National Library of Canada
Bibliothèque nationale du Canada

Canadian Theses Service
Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

NL-339 (r. 88/04)c

Canada

UNIVERSITY OF ALBERTA

CONTROL OF REDUNDANT MANIPULATORS

BY

Philip Ka-Yip Pang (C)

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

CONTROL SYSTEMS

DEPARTMENT OF ELECTRICAL ENGINEERING

EDMONTON, ALBERTA

SPRING 1991

National Library of Canada
Bibliothèque nationale du Canada

Canadian Theses Service
Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-66699-4

Canada

UNIVERSITY OF ALBERTA
RELEASE FORM

NAME OF AUTHOR:	Philip Ka-Yip Pang
TITLE OF THESIS:	Control of Redundant Manipulators
DEGREE:	Master of Science
YEAR THIS DEGREE GRANTED:	Spring, 1991

PERMISSION IS HEREBY GRANTED TO THE UNIVERSITY OF ALBERTA LIBRARY TO REPRODUCE SINGLE COPIES OF THIS THESIS AND TO LEND OR SELL SUCH COPIES FOR PRIVATE, SCHOLARLY OR SCIENTIFIC RESEARCH PURPOSES ONLY.

THE AUTHOR RESERVES OTHER PUBLICATION RIGHTS, AND NEITHER THE THESIS NOR EXTENSIVE EXTRACTS FROM IT MAY BE PRINTED OR OTHERWISE REPRODUCED WITHOUT THE AUTHOR'S WRITTEN PERMISSION.

Philip Pang

(Student's Signature)

Permanent Address:

8D, FU BOU COURT
32, FORTRESS HILL ROAD,
NORTH POINT,
HONG KONG

DATE : February 15, 1991

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

THE UNDERSIGNED CERTIFY THAT THEY HAVE READ, AND RECOMMEND TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH FOR ACCEPTANCE, A THESIS ENTITLED **CONTROL OF REDUNDANT MANIPULATORS** SUBMITTED BY **PHILIP KA-YIP PANG** IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF **MASTER OF SCIENCE** IN ELECTRICAL ENGINEERING IN CONTROL SYSTEMS.

Dr. V.G. Gourishankar
(Supervisor)

Dr. R.E. Rink

Dr. Hong Zhang

DATE : February 15 1991

Dedication

In memory of my father

and

in gratitude to my mother

ABSTRACT

A robot manipulator operating in three dimensional workspace should have six degrees of freedom for the end effector to be capable of reaching every point in the workspace. Such a robot is said to be kinematically non-redundant. By implication, a kinematically redundant robot has more than the minimal number of degrees of freedom required to reach points in the workspace. Redundancy can be used to achieve goals that cannot be achieved by a non-redundant manipulator. In this thesis a time-optimal control algorithm is developed for three link planar manipulator operating in an obstacle-free two dimensional workspace. The solution depends on the following factors : The erd effector trajectory is specified, the initial configuration of the manipulator is known and the pseudoinverse of the Jacobian is used to obtain the inverse kinematic solution at each step along the specified trajectory. The acceleration of the end effector along a specified trajectory is used as the control variable. This minimum time algorithm is extended to avoid an obstacle in the workspace. The solution is dependent on the joint trajectories which are determined a priori for a given obstacle. Simulation results are presented.

ACKNOWLEDGEMENTS

I would like to give thanks and honor to my Sovereign LORD for granting me wisdom for this research project. I would also like to express my deepest gratitude towards my supervisor Professor V.G. Gourishankar for his encouragement, support and guidance in the past two years. It is in him that I found a teacher, friend and inspiration. For this, I can never express enough gratitude.

The financial support by the Electrical Engineering Department, University of Alberta, is gratefully acknowledged, for without its funding this work would not have been possible. Throughout the preparation of this work, I have enjoyed the atmosphere and collaboration of the control group in this department, in particular, I would like to thank Frank Niscak, Dr. Joel King and Eugene Cao for their precious advice and help.

I would be forever grateful to my special and beloved friend, Esther Poon, for her tender love and care. She has given me the moral support that I would ever need and shared with me in times of difficulties and in moments of pressure and stress. A sincere special thanks must also be extended to my sister Carol for her patience, continual encouragement and prayer support.

Last, and most important, I am in debt to the the brothers and sisters in the Edmonton Chinese Alliance Church for their love, understanding, encouragement and prayer support, especially Rev. Titus Yu from whom I have benefited immensely from his teaching and preaching.

TABLE OF CONTENTS

1. Introduction 1

1.1 Robotics Background 1

1.2 Limitations of Non-Redundant Manipulator 3

1.2.1 Obstacles in Workspace 3

1.2.2 Singularity 4

1.3 The Use of Redundant Manipulator 5

1.4 Objective of This Thesis 9

1.5 Structure of Thesis 10

2. Mathematical Background of Redundant Manipulators 12

2.1 Introduction 12

2.2 Range Space and Null Space of a Matrix 13

2.3 Definition of the Pseudoinverse 14

2.4 Theorems and Properties of the Pseudoinverse 16

2.5 Application of the Preceding Results 21

3. Kinematics and Dynamics of a Three Link Planar Redundant Manipulator 22

3.1 Introduction 22

3.2 Manipulator Kinematics 22

3.3 Manipulator Dynamics 25

4. Time Optimal Control of A Redundant Manipulator 32

4.1 Introduction 32

4.2 Time Optimal Control of Non-Redundant Manipulators 34

4.3 Time Optimal Control of Redundant Manipulators -

Problem Formulation 42

4.4 Solution to the Time Optimal Problem for the Redundant Case 44
4.4.1 Single Switching Case 45
4.5 Computational Problems 48
4.5.1 Integration Technique 48
4.5.2 Extrapolation Methods 49
4.5.3 Generation of the Zero-th Column 52
4.6 Examples 54
4.7 Multiple Switching Case 66
5. An Obstacle Avoidance Algorithm 69
5.1 Introduction 69
5.2 Inverse Kinematic Function Approach 69
5.2.1 Formulation of the Additional Function 70
5.2.2 Inverse Kinematic Solutions 73
5.2.2.1 Choice of Obstacle Avoidance Constants 76
5.2.3 Example 83
5.3 Incorporation of the Obstacle Avoidance Algorithm 83
5.3.1 Example 88
6. Computer Implementation 94
6.1 Introduction 94
6.2 Program Structure 94
6.2.1 Time Optimal Algorithm 95
6.2.2 Obstacle Avoidance Algorithm 98
6.3 Simulation and Results 98
7. Conclusions 104
Bibliography 106

LIST OF TABLES

Table 3.1 Parameters of the three link redundant manipulator 31

Table 4.1 Numerical values of the second example 66

LIST OF FIGURES

Figure 1.1	A Puma 560 robot	2
Figure 1.2	A singular position for non-redundant manipulator	4
Figure 1.3	A non-redundant manipulator and an obstacle	7
Figure 1.4	A redundant manipulator and an obstacle	8
Figure 3.1	A three link redundant manipulator	23
Figure 4.1	A two link planar manipulator	35
Figure 4.2	Two possible configurations for a 3-link planar redundant manipulator	42
Figure 4.3	Time optimal trajectory for the single switching case	46
Figure 4.4	End effector motion	55
Figure 4.5	Phase plane trajectory and maximum velocity curve	56
Figure 4.6	The resulting a)Joint angles b)Joint velocities and c)Joint torques for the end effector motion	57
Figure 4.6	(b)	58
Figure 4.6	(c)	59
Figure 4.7	End effector motion for the second example	61
Figure 4.8	Phase plane trajectory for the second example	62
Figure 4.9	The resulting a)Joint angles, b)Joint velocities and c)Joint torques for the second example	63
Figure 4.9	(b)	64
Figure 4.9	(c)	65
Figure 4.10	Multiple switching curve	67
Figure 5.1	Effect of s on h(a,s)	79

Figure 5.2 Effect of a on h(a,s) 80
Figure 5.3 Intermediate result of obstacle avoidance example 84
Figure 5.4 Final result of obstacle avoidance example 85
Figure 5.5 Phase plane trajectory for the time optimal obstacle example..... 90
Figure 5.6 The resulting a)Joint angles, b)Joint velocities and c)Joint torques for the time optimal obstacle example 91
Figure 5.6 (b) 92
Figure 5.6 (c) 93
Figure 6.1 Simulation Result of Figure 4.4 101
Figure 6.2 Simulation Result of Figure 4.7 102
Figure 6.2 Simulation Result of Figure 5.4 103

Chapter 1

Introduction

1.1 Robotics Background

Early research work on robotics can be traced back to the late 1940's when remotely controlled master-slave mechanical manipulators were developed in the U.S.A to handle radioactive materials. The first industrial manipulator with computer control was introduced in 1959 by Unimation Inc. This was the predecessor to the present-day popular PUMA robot (Fig. 1.1). Robotics research has come a long way during the past three decades. Many sophisticated manipulators with more sophisticated control schemes have been developed in recent years in the laboratories and some of these have found their way into practical real world applications.

For a general and typical industrial application, in three dimensional workspace, a manipulator will have to have as a minimum six degrees of freedom and the robot shown in Fig. 1.1 belongs to this category. Questions relating to the design and control of such a robot to perform desired tasks are being tackled by the research community. One such question is related to the number of degrees of freedom which the manipulator possesses. It has been shown that a manipulator with more than the minimum degrees of freedom (a redundant manipulator) has more flexibility and will be necessary in some situations. This thesis is concerned with the role of redundancy in robot manipulators. A more specific statement of the thesis objective will be given later in

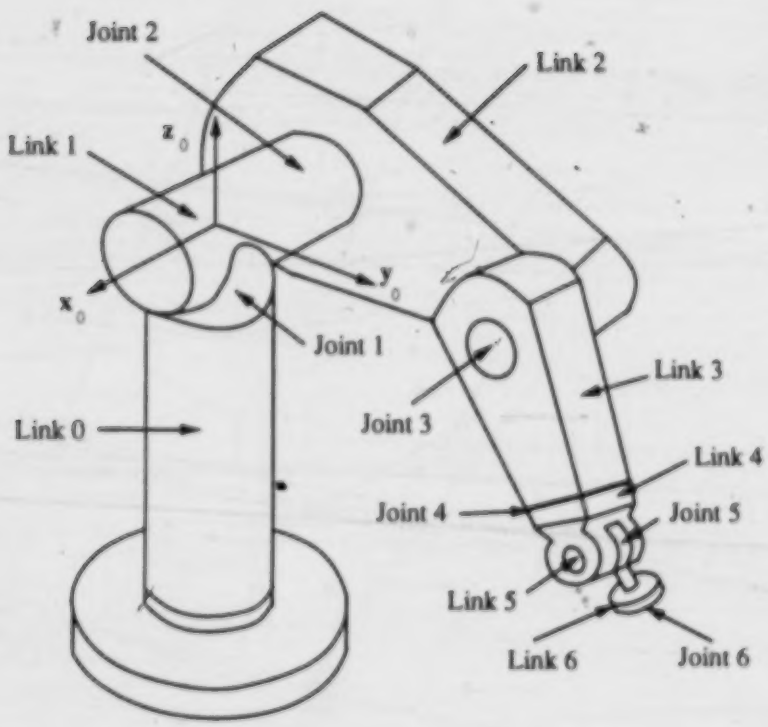

Figure 1.1 A Puma 560 robot

this chapter.

1.2 Limitations of Non-Redundant Manipulators

A manipulator with the minimum degrees of freedom for a given workspace is called a non-redundant manipulator The presence of obstacles in the workspace and singularity configurations (to be defined later) of the manipulator are the two major limitations of a non-redundant manipulator. A brief explanation of these two limitations is given in the following subsection. A significant portion of the research effort during the past decade has addressed these two limitations.

1.2.1 Obstacles in Workspace

In practical applications, it is highly unlikely that the workspace of a manipulator will be free from obstacles. Suppose the manipulator is to move from one point to another point in the workspace and there are no restrictions to the joint motions, the trajectory is usually determined by the control scheme used. An object can be obstructing the manipulator in two different circumstances. The first occurs when an obstacle is in the path of the end effector as it moves along the trajectory. The links are not affected as they move along their respective trajectories corresponding to the movement. The second case occurs when obstacles are encountered by the links of the manipulator as they move when the end effector moves from an initial position to a final position.

Obstacle avoidance can be achieved relatively more easily at the path planning stage for the first case. Typically, the control algorithm is constructed in such a way that the manipulator is made to follow a predetermined path from the specified initial point to the specified end point avoiding the obstacle in the work space if such a path is available. This usually results in additional traveling distance for the end effector

and additional computations are to search for the alternate path. Artificial intelligent algorithms are often used to find the obstacle avoidance path efficiently and effectively.

The determination of the obstacle avoidance algorithm in the second case is usually more difficult. More than one intermediate link may run into the obstacle as the arm moves along the specified trajectory. The obstacle avoidance algorithm which can tackle this problem is therefore more complicated. Moreover, there is always the possibility that an obstacle free trajectory may not exist.

There are situations where the end effector *trajectory* is just as important as the end points themselves or the end effector must pass through certain points in the workspace. It is impossible to find any alternate path to avoid the obstacle with a non-redundant manipulator. In such a case, a redundant manipulator can be used to solve the problem.

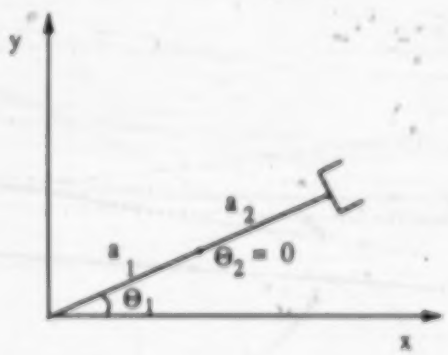

Figure 1.2 A singular position for non-redundant manipulator

1.2.2 Singularity

Another difficulty encountered by robot motion algorithms is the singular configuration of a manipulator. All manipulators have locations in the workspace

where movement or rotation in at least one direction cannot be achieved. These specific locations are known as the singularities of the manipulator. In Figure 1.2, the end effector cannot move in the direction parallel to the link a_1. Near such configurations, small displacements of the end effector require excessively large and physically unrealizable joint speeds, which introduce severe inaccuracies in the desired motions.

Singularities are divided into two categories. One is the workspace boundary singularity, which typically occurs as the arm is fully stretched out or folded back when the end effector is at the boundary of the workspace. The other singularity is known as the workspace interior singularity. It is found within the workspace and is usually caused by the lining up of two or more joint axes of the manipulator. Obviously, the boundary singularity can never be eliminated since there must be a boundary somewhere in the workspace. The interior singularity, however, can be avoided and there are two traditional approaches to deal with this well known problem. The first method avoids these singular regions completely in the path planning stage. Envelopes enclosing the singularities are identified and motions in the state space are restricted to lie outside these envelopes. The other approach, however, is to compromise between the desired end effector motion and possible joint motion. Velocity constraints are placed on the end effector to prevent excessive velocities and joint angle trajectories are interpolated between singular points. The major drawback of this scheme is that the envelopes practically eliminate one or more degrees of freedom of the manipulator. Researchers have turned their attention to see how an extra degree of freedom can be employed to resolve these difficulties.

1.3 The Use of a Redundant Manipulator

A manipulator, with more than the minimum number of degrees of freedom necessary for a particular task in a given workspace is known as a kinematically redundant manipulator. The number of extra degrees of freedom is referred to as the redundancy of the manipulator. It has also been suggested that future general purpose manipulators should contain at least one degree of redundancy [Hollerbach 84]. Only a few redundant manipulators have actually been built. Most of the research has been device-independent. Among those are the seven degrees of freedoms tendon-driven, torque controlled robot [Takase, Inoue, and Sato 74], the articulated seven degree of freedoms UJIBOT driven by d.c. servomotors equipped with tachogenerator at each joint to measure the joint angle and velocity [Hanafusa, Yoshikawa and Nakamura 81], an 8 degree-of-freedom sheep shearing robot [Trevelyan, Kovesi, and Ong, 83] and a seven degrees of freedom CESAR research manipulator with a spherical wrist. (Dubey 87) Most of the theories of the redundant manipulator have been developed by simulations.

Until recently research work in redundant manipulators has been concentrated on robots with only one degree of redundancy. One degree of redundancy is, in general, sufficient to avoid singularities and avoid obstacles, if it is added to a non-redundant manipulator effectively. Consider the specified path in Figure 1.3 and Figure 1.4 for the same motion from point A to point B of a non-redundant manipulator and a redundant manipulator. Both manipulators have the same total link lengths and the link length for all the links is the same. Suppose the task requires the end effector to follow a straight line motion from A to B, and an obstacle, (represented by the rectangular box) is presented in the workspace. It can be seen in Figure 1.3 that the non-redundant manipulator collides with the obstacle before it completes the specified path. However, the redundant manipulator manages to follow the specified trajectory without running into the obstacle. For the specified path, the end effector is not obstructed by

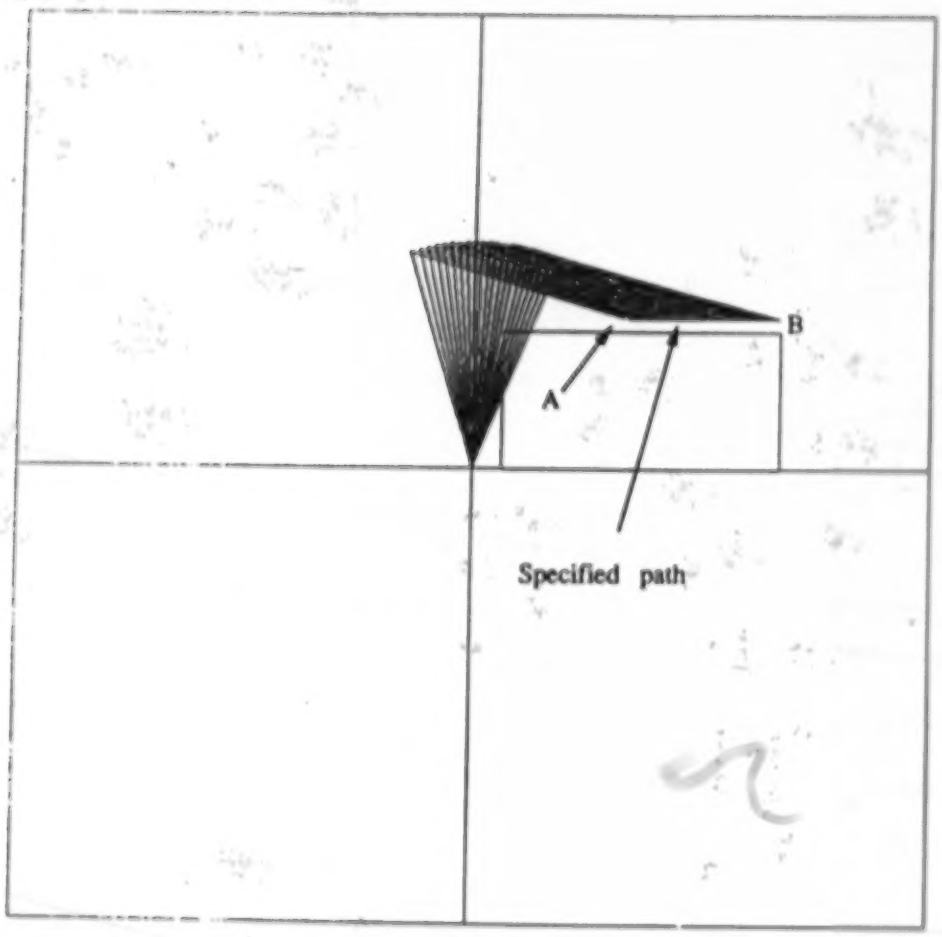

Figure 1.3 A non-redundant manipulator and an obstacle

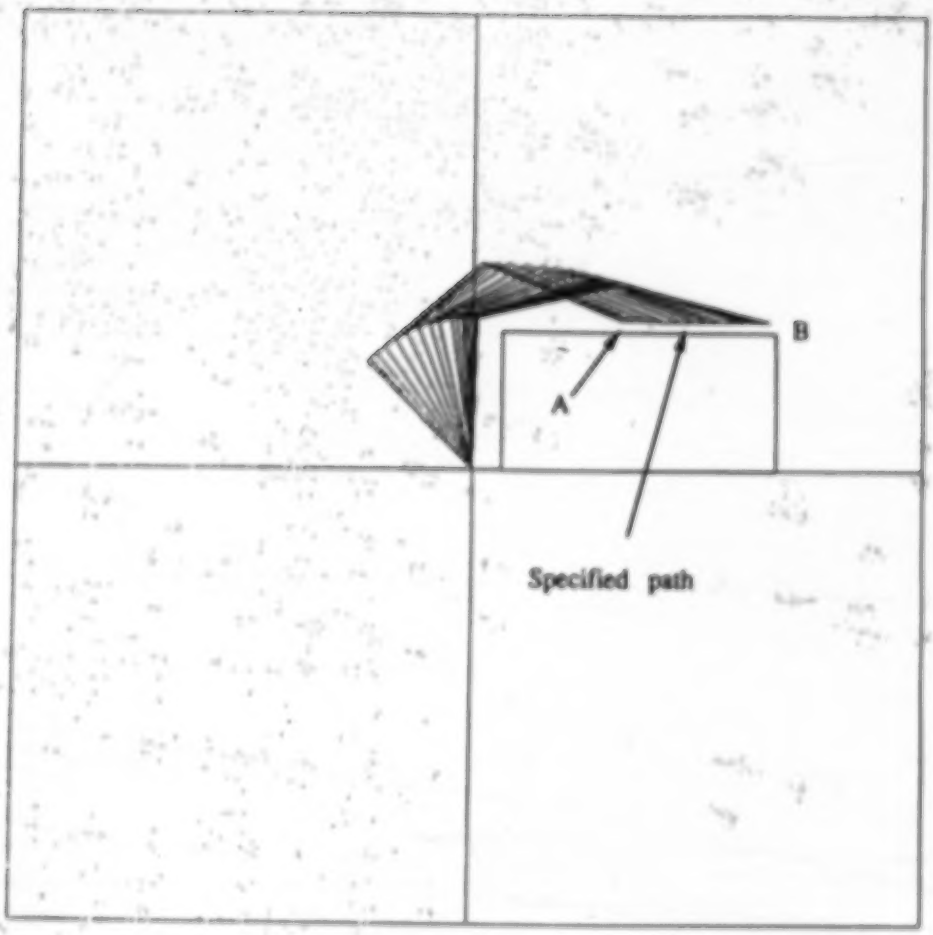

Figure 1.4 A redundant manipulator and an obstacle

obstacles, but rather the intermediate link may run into obstacles because of the confined workspace as shown in this example. This simple example demonstrates the necessity for the use of a redundant manipulator for obstacle avoidance.

1.4 Objective of This Thesis

For many present day applications, the speed of robotic manipulators is rather slow when compared to the human hand. Current manipulator controllers are not designed to move the arm as fast as possible. The maximum velocity and acceleration are actually obtained by trial and error until the actuators are saturated. For higher productivity, it is desirable to operate the manipulator at the maximum allowable velocity at all points along the trajectory. Time optimal algorithms have been developed in the past for non-redundant manipulators. The objective of these algorithms is either to find the minimum time path for the end effector to move from one point to another point in the workspace [Geering et el. 86] or to move the end effector along a specified path in minimum time [Bobrow et el. 83].

Instead of the former time optimal control approach which uses the joint torques as the control variables. Bobrow uses a single control variable namely, the acceleration of the end effector along the planned trajectory to obtain the minimum time control. In doing so, the whole trajectory in the Cartesian space can be specified in contrast to only the end points for most of the other time optimal control algorithms. Torque constraints are placed and transformed into the velocity limit curve in the phase plane of the control variable. The minimum time is achieved by driving the end effector along the trajectory at maximum acceleration and then at maximum deceleration. One switching in the phase plane is required as long as the switching curve remains below the maximum velocity curve. However, a more complicated algorithm for multiple switching is needed when the joint torques are not sufficient in meeting the velocity

requirement.

In this thesis, non-redundant time optimal control strategy of Bobrow above is extended to the redundant case. The redundant time optimal algorithm is further extended to include obstacle avoidance. For non-redundant manipulators, it is only possible to achieve either minimum time specified paths [Bobrow 83] or minimum time obstacle avoidance [Dubrowsky 86]. In this thesis, both path specification and obstacle avoidance are include in one time optimal control algorithm. This is made possible by utilizing the extra degree of freedom in redundant manipulators. This algorithm is verified experimentally by means of a simulation program written in C on the Unix system. Graphical animation is also written in C using the Computer Graphics Interface (CGI) available in the Sun workstation.

Time optimal control for redundant manipulators, however, has difficulties with the non-uniqueness of the intermediate joint trajectories because there are infinitely many possible sets of joint angles that can satisfy an end effector requirement. Therefore, the time optimal algorithm developed in this thesis is optimal only under the condition that the joint angles are specified. In this thesis, the starting angles are assumed to be given and the intermediate joint trajectories are obtained from the pseudoinverse. The minimum time referred to in the following chapters is only optimal under the constraint of the pseudoinverse inverse kinematic solution.

1.5 Structure of Thesis

This thesis is weighted equally in both the theoretical and experimental aspects. The background and some theoretical results for the time optimal control algorithm are presented. These are followed by the computer simulations and experimental results. A more detail description of the objective of each chapter is given at the beginning of each chapter. Only a brief outline is mentioned here.

Chapter two reviews the mathematical backgrounds for redundant manipulators. It is intended to fill in the mathematical details required to understand the control algorithms described in later chapters. Major properties for underdetermined systems and the null space are given. The use and definitions of the pseudoinverse are also introduced.

Chapter three presents the three link redundant manipulator model that is used later to implement the time optimal algorithm. The dynamic and kinematic models of the manipulator are derived. Assumptions and parameters of the manipulator are also given.

Chapter four develops the time optimal control strategy for a redundant manipulator. It is a three link planar manipulator operating in two dimensions. The use of the pseudoinverse to obtain a unique set of joint angles for a Cartesian position is discussed. Single switching cases as well as the more complicated multiswitching cases are both considered.

Chapter five develops an obstacle avoidance algorithm. This avoidance algorithm is applied to the time optimal control developed in chapter 4 to avoid obstacle in the workspace.

Chapter six describes the programming structure of the algorithms in chapter four and five. The control laws are applied to a computer simulated three link planar redundant manipulator driven by a set of input torques. The torques calculated in chapter 3 and 4 are used as inputs to the program to produce the expected time optimal motions. Examples are given to illustrate the developed theories.

Chapter seven states the conclusions and suggests future research topics.

Chapter 2

Mathematical Background of Redundant Manipulators

2.1 Introduction

In the case of a non-redundant manipulator, the kinematic equation which relates the joint velocities to the Cartesian velocities is written as

$$J\dot{\underline{\theta}} = \dot{\underline{x}} \tag{2.1}$$

where $\dot{\underline{\theta}}$ and $\dot{x}$ are the vector of the joint velocities and the vector of the Cartesian velocities respectively. J is the Jacobian matrix of the kinematic equation which contains trigonometric functions of the joint angles. For non-redundant manipulators, the sizes of the vectors $\dot{\underline{\theta}}$ and $\dot{\underline{x}}$ are the same because the number of joints of the manipulator is the same as the degrees of freedom. Therefore, the Jacobian is a square matrix. However, for the redundant case, the dimension of $\dot{\theta}$ is greater than the dimension of $\dot{x}$. The Jacobian matrix is, therefore not square and when the joint velocities are to be expressed in terms of the Cartesian velocities, the inverse of a non-square matrix is required. The "inverse" of a non-square matrix is known as the pseudoinverse. The derivation of kinematic equation of a redundant robot relies heavily on the pseudoinverse and its associated matrix properties. These properties are not only useful in obtaining the mathematical model but are also essential to most of the present

control algorithms for the redundant manipulators. In this chapter, the definitions and theorems relevant to the pseudoinverse of a matrix will be reviewed.

2.2 Range Space and Null Space of a Matrix

For a general matrix H with m rows and n columns, the range space and null space are defined below.

Definition 2.1

The range space of a matrix H, denoted by $R(H)$, is defined as a set of vectors such that

$$R(H) = \{ z : z = Hx \} \tag{2.2.1}$$

where x is an arbitrary vector in the Euclidean space that serves as the domain of H.

The dimension of the range space is sometimes referred to as the rank of the matrix. For robotic manipulators, the range space of the Jacobian matrix, $R(J)$, is called the manipulatable space and the dimension of $R(J)$ is called the degree of manipulatability (d.o.m). Since the Jacobian matrix is joint angle dependent, d.o.m is defined for each geometric position of the manipulator. The singularity region of the workspace will then have a lower degree of manipulatability.

Definition 2.2

The null space of H, denoted by $N(H)$, is a set of vectors that maps to zero through H such that

$$N(H)=\{ x \ : \ Hx = 0 \} \tag{2.2.2}$$

where x is an arbitrary vector in the Euclidean space that serves as the domain of H.

The dimension of the null space, namely the nullity, is then the number of independent vectors that span the null space. For redundant manipulators, the null space of the Jacobian matrix, $N(J)$, is called the redundant space and its nullity is called the degree of redundancy (d.o.r). From the above definition, it is noted that all the vectors, x in the null space must be orthogonal to the row vectors of H if they were to map to zero by H. Therefore, the null space of a matrix H must be composed of vectors which are also linearly independent to the row vectors which defines the range space. For a non-square matrix H, when the number of columns is larger than the number of rows, the dimension of the null space of H is the same as the difference between the columns and the rows. Denote the rank of an $m \times n$ matrix H by $\gamma(H)$ and its nullity by $\rho(H)$. The nullity and rank of a matrix are related by

$$\rho(H) + \gamma(H) = n \tag{2.3}$$

which implies the sum of the dimension of the null and range space of a redundant manipulator is equal to the number of columns of the Jacobian matrix. Therefore, the sum of d.o.m. and d.o.r is actually the total number of joints of the manipulator.

2.3 Definition of the Pseudoinverse

The use and definitions for pseudoinverse arise from the need to take the inverse of a non-square matrix. It is also called generalized inverse for it is virtually the inverse of a general matrix. It is extensively used to find a solution for an underdetermined system of linear equations out of the infinitely many possibilities. The pseudoinverse of a rectangular matrix can be defined by the following theorem.[Albert 72]

Theorem 2.1

For any matrix H, its pseudoinverse defined as

$$H^{+} = \lim_{\delta \to 0} (H^T H + \delta^2 I)^{-1} H^T \tag{2.4.1}$$

$$H^{+} = \lim_{\delta \to 0} H^T (H H^T + \delta^2 I)^{-1} \tag{2.4.2}$$

always exists. It can also be written as

$$H^{+} = H^T (H H^T)^{-1} \tag{2.4.3}$$

if the rows of H are all linearly independent and as

$$H^{+} = (H^T H)^{-1} H^T \tag{2.4.4}$$

if the columns of H are all linearly independent.

For an $n \times m$ rectangular matrix, it is at its full rank if and only if either its rows or columns have all linearly independent vectors. Therefore, an n dimensional vector space can have at most n linearly independent vectors, and definition (2.4.3) is used only when the row dimension is less than the column dimension, otherwise definition (2.4.4) would be used. It is an underdetermined system in the case of the redundant manipulator, and the pseudoinverse of the Jacobian is defined according to (2.4.3).

The number of independent vectors defines the dimension of the range space of a matrix, and its transpose cannot have more independent vectors than itself. The range space of a matrix, $R(H)$, is actually the range space of HH^T. This means that HH^T is singular if and only if the rows of H are not all linearly independent. This concludes that the pseudoinverse defined in (2.4.3) always exists if the rectangular matrix, H, is at its full rank. Definition (2.4.4) can be shown to exist with the same arguments.

It is seen from equation (2.4.3), that the pseudoinverse is indeterminate whenever HH^T is singular. In the application of the pseudoinverse to the redundant manipulator, the Jacobian does have linearly dependent rows when the end effector is at the boundary of the workspace. These points define the singular regions of the redundant

manipulator.

2.4 Theorems and Properties of the Pseudoinverse

Pseudoinverse is sometimes called the Moore-Penrose pseudoinverse for Penrose's contribution in his paper in 1955 [Penrose 1955]. He brought the interest of the pseudoinverse back by pointing out that the pseudoinverse provides an unique solution to a set of matrix equations and that it satisfies the following Penrose conditions. For any matrix H, and its pseudoinverse $B=H^+$, they must satisfy

$$HBH = H \tag{2.5.1}$$

$$BHB = B \tag{2.5.2}$$

and both HB and BH are symmetric.

Another way of defining the pseudoinverse would be to make use of the well known singular value decomposition theorem (SVD). A better and more complete definition to the pseudoinverse can be derived directly from the Penrose conditions and the SVD theorem. Not only does the SVD help to understand the concept in behind, it can also be extended easily to give a numerical computation algorithm that has substantially higher accuracy than (2.4.3).

Theorem 2.2

For an $m \times n$ matrix H, let L be an $r \times r$ diagonal matrix of (HH^T)'s non-zero eigenvalues λ's arranged in any order, then there exists an $m \times r$ matrix P and an $r \times n$ matrix Q which satisfies the following conditions

$$H = PL^{1/2}Q \tag{2.6.1}$$

$$HH^T = PLP^T \tag{2.6.2}$$

$$HH^+ = PP^T \tag{2.6.3}$$

$$P^T P = I \tag{2.6.4}$$

$$H^T H = Q^T L Q \tag{2.6.5}$$

$$H^+ H = Q^T Q \tag{2.6.6}$$

$$QQ^T = I \tag{2.6.7}$$

where

$$L^{1/2} = diag\,(\lambda_1^{1/2}, \lambda_2^{1/2}, \ldots, \lambda_n^{1/2}) \tag{2.6.8}$$

Equation (2.6.1) of Theorem 2.2 is referred to as the SVD theorem. Any matrix H can be decomposed into product of three matrices, the diagonal eigenvalues matrix L and its corresponding orthogonal eigenvector matrices P and Q. From (2.6.4) and (2.6.7), it is seen that the columns of P and the rows of Q are orthogonal. Equations (2.6.2) and (2.6.5) indicates that the column of P and the row of Q consist of the eigenvectors of HH^T and $H^T H$ respectively. Due to the fact that any matrix of the form HH^T has non-negative eigenvalues, the square root of L defined in (2.6.8) is always real and positive. The size of the diagonal matrix L is governed by the number of non-zero eigenvalues of HH^T or the rank of H.

Starting from equation (2.6.1), with the help of properties (2.6.2) to (2.6.7) and the Penrose conditions, the pseudoinverse of H can be defined in a similar decomposed form. Substitute equation (2.6.1) into (2.6.3) yields

$$PL^{1/2}QH^+ = PP^T \tag{2.7.1}$$

$$\Rightarrow \qquad L^{1/2}QH^{+} = P^{T} \tag{2.7.2}$$

Since the inverse of a diagonal matrix is simply the reciprocal of its diagonal elements, $L^{-1/2}$ is denoted as the inverse of $L^{1/2}$. Multiplying equation (2.7.2) with $Q^{T}L^{-1/2}$ gives

$$Q^{T}QH^{+} = Q^{T}L^{-1/2}P^{T} \tag{2.7.3}$$

Finally, substituting (2.6.6) into the left side of (2.7.3) and applying the Penrose condition (2.5.2), the pseudoinverse can be defined as

$$H^{+}HH^{+} = Q^{T}L^{-1/2}P^{T}$$

$$H^{+} = Q^{T}L^{-1/2}P^{T} \tag{2.7.4}$$

It is obvious that in equation (2.7.4) the linear independent vector restriction in (2.4.3) is released. The rank of H, which is r, affects only the dimensions of P, Q and L, and (2.7.4) holds regardless. For numerical computation, it is more convenient to break down the matrix equation (2.7.4) into a sum of vector products. It would be much more difficult if L is not diagonal. Making use of this advantage, the pseudoinverse can be rewritten as

$$H^{+} = \sum_{i=1}^{k} \lambda_i^{-1/2} q_i^{T} p_i \tag{2.8}$$

where q_i and p_i are the corresponding row vectors of λ_i for matrices P and Q respectively.

From the definitions given above, it is easily seen that the solution provided by the pseudoinverse is unique. Sometimes, the pseudoinverse solution may not be a

desirable one among all other possibilities. The pseudoinverse would not be appreciated as much if there is no other means to look for the wanted solution. The null space of the pseudoinverse provides a means to find all the possible solutions to the underdetermined system.

For any vector x in the finite dimension Euclidean space that has a linear subspace L, there exists an unique vector $\hat{x} \in L$ that has the property that x and $x-\hat{x}$ are orthogonal. This implies that x has an unique decomposition

$$x = \tilde{x}+\hat{x} \tag{2.9}$$

where $\tilde{x}$ is orthogonal complement of subspace L. Now, consider the projection of this vector on the range space of a matrix $H^T H$, then $\hat{x}$ and $\tilde{x}$ will belong to $R(H^T H)=R(H^T)$ and $N(H^T H)=N(H)$ respectively. The projection vector on the null space of H can be written as

$$\tilde{x} = x - \hat{x} \tag{2.10}$$

It can be shown that the projection of a vector on $R(H^+H)$ is the same as the projection on $R(H^T)$, therefore it follows from (2.10)

$$x-\hat{x}$$

$$\Rightarrow x-H^+Hx$$

$$\Rightarrow (I-H^+H)x \tag{2.11}$$

The matrix $(I-H^+H)$ is known as the null space projection matrix for it is the afterimage of the vector x onto the null space of H. This null space projection matrix is frequently used to obtain a solution for a homogeneous system of equations, $Hx=0$. When H is at full rank, the null vector is the only solution, therefore the null space

projection matrix must be a zero matrix. From (2.11), it implies that H^+H is an identity matrix. Moreover, this projection matrix is used extensively in conjunction with the pseudoinverse to yield a complete solution for a non-homogeneous system. The major properties of the pseudoinverse can be summerized by the virtues of the following theorem

Theorem 2.3

a) x_0 minimizes

$$\|z - Hx\|^2 \tag{2.12.1}$$

if and only if x_o is of the form

$$x_0 = H^+z + (I - H^+H)y \tag{2.12.2}$$

for some y.

b) The value of x which minimizes (2.12.1) is unique if and only if

$$H^+H = I \tag{2.12.3}$$

Equation (2.12.3) is true if and only if zero is the only null vector of H.

c) The equation

$$Hx = z \tag{2.12.4}$$

has a solution if and only if

$$HH^+z = z \tag{2.12.5}$$

Equation (2.12.5) is true if and only if $z \in R(H)$. x_0 is a solution to (2.12.4) if and only if it is of the form of (2.12.2). Equation (2.12.4) has an unique solution ($=H^+z$) if and only if $HH^+z=z$ and $H^+H=I$.

The proof of this theorem is available in [Albert 72].

2.5 Application of the Preceding Results to Redundant Manipulator

Theorem 2.3 provides much insight to the use of the pseudoinverse and the null space in general. Part (c) of this theorem is directly applicable to the redundant manipulator for (2.12.4) and it is of the same form as the kinematic Jacobian equation (2.1). Equation (2.12.5) states the only criterion for the Jacobian equation to have a valid solution, that is when $\dot{x}$ lies in the range space of J. This is always true unless the rank of the Jacobian matrix is less than the dimension of the vector $\dot{x}$. In the case when J is deficient, the Jacobian equation can only be satisfied if the dimension of $\dot{x}$ is reduced accordingly. This explains the reduction of the degree of freedom at singularity.

From part (b), the joint angle solution for redundant manipulators is unique only if zero is the only vector in the null space of the Jacobian. Since the null space of the Jacobian has at least one non-zero vector, there are always infinitely many possible joint velocity vectors that satisfy a particular Cartesian velocity vector. The null space projection in (2.12.2) provides a mean to find the desired solution through y. From (2.12.2), it is also seen that the pseudoinverse solution without the null space is of minimum norm because the general solution with the addition of a null space projection vector to the pseudoinverse can only increase the norm of x_0.

The first part of Theorem 2.3 is perhaps the most useful for it has no restriction on H. The minimum of the norm of (2.12.1) is obviously zero when H has full rank, however when H is singular, x_0 simply gives the minimum norm solution to (2.12.1). The minimum norm of (2.12.1) is virtually the minimum error of (2.12.4). Theorem 2.3 simply provides the kinematic Jacobian equation with an exact solution whenever it is available and a minimum error solution if there is no exact answer.

Chapter 3

Kinematics and Dynamics of a Three Link Planar Redundant Manipulator

3.1 Introduction

In this thesis, all the theoretical ideas and control algorithms are developed for a three link manipulator shown in Fig 3.1. Since the motion is confined to a plane, the three link manipulator can be considered as a redundant manipulator.

The kinematics and dynamics of the manipulator are derived in this chapter. The end effector is assumed to have insignificant mass, hence the dynamics of the end effector are ignored. The orientation of the end effector is not taken into account for the ease and simplification of the derivation. The orientation of the end effector for planar manipulator applications is usually fixed at a certain angle and the algorithm presented in this work is independent of the orientation of the end effector.

3.2 Manipulator Kinematics

The link length and joint angle variables used in the following discussion refer to those in Figure 3.1. Kinematics is the representation of the end effector position in terms of the joint angles in the two dimensional plane. To simplify expressions, trigonometric functions like sine and cosine are denoted by s and c respectively, and the appended subscripts imply the trigonometrical operation on the sum of the corresponding joint angles. For example, $s_{12}=sin(\theta_1+\theta_2)$. Now, the Cartesian position (x,y) of

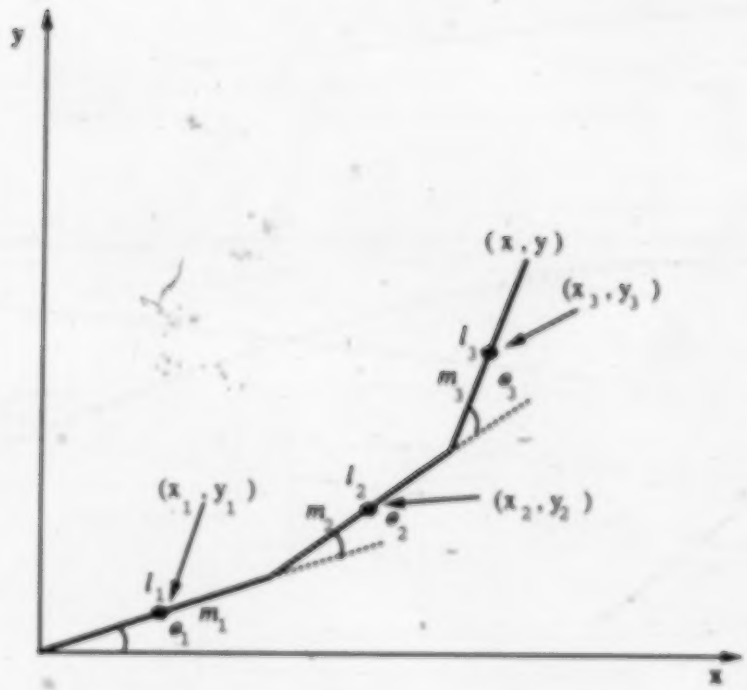

Figure 3.1 A Three Link Redundant Manipulator

the end effector in the plane can be expressed as

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} l_1c_1 + l_2c_{12} + l_3c_{123} \\ l_1s_1 + l_2s_{12} + l_3s_{123} \end{bmatrix} \tag{3.1.1}$$

With the above equations, any Cartesian position is uniquely defined, given a set of joint angles. Solving (3.1.1) for the θ's, given any (x,y) position in the plane is known as the inverse kinematics of the manipulator.

To analyze the motion of the manipulator, the angular velocities of the joints are required. The description of the manipulator motion in the joint space in terms of the motion in the Cartesian space is obtained from the kinematic equation (3.1.1). Taking the time derivative of (3.1.1), the relation of the joint and Cartesian velocity is given as follows

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} -l_1s_1-l_2s_{12}-l_3s_{123} & -l_2s_{12}-l_3s_{123} & -l_3s_{123} \\ l_1c_1+l_2c_{12}+l_3c_{123} & l_2c_{12}+l_3c_{123} & l_3c_{123} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} \tag{3.1.2}$$

where $(\dot{x},\dot{y})$ is the velocity in the Cartesian plane and $\dot{\theta}_i$ is the velocity of the i^{th} joint. The above equation can also be written in matrix form as

$$\dot{\underline{x}} = J(\theta)\dot{\underline{\theta}} \tag{3.1.3}$$

where $\dot{\underline{\theta}}$ is the joint velocity vector and $\dot{\underline{x}}$ is the Cartesian velocity vector. $J(\theta)$ is a 2×3 matrix known as the Jacobian matrix of the manipulator. Since the Jacobian is a non-square matrix, the joint velocities can be obtained from the pseudoinverse as

$$\dot{\underline{\theta}} = J^{+}(\theta)\dot{\underline{x}} \tag{3.1.4}$$

3.3 Manipulator Dynamics

It is important to know the forces or torques and the related equations of motion if any control is to be done to the arm. The dynamic model of the manipulator is usually very complicated and highly non-linear. The non-linearities arise from the Coriolis and centrifugal forces. Gravitational force also contributes to the difficulty in deriving the dynamic equations of the manipulator. The dynamic equations of the manipulator are the expressions of the joint torques in terms of the joint angles, velocities and accelerations. Formulation of the dynamics is far more complicated than that of the kinematics. The Lagrangian approach is used instead of the numerical iterative Newton-Euler dynamics formulation because the redundant manipulator used here has a relatively small number of degrees of freedom. Closed form dynamic expressions are derived below to reduce computation time for the evaluation of the inertia matrix, the Coriolis and centrifugal forces. The Lagrange-Euler equation is given by

$$\tau_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}_i}\right) - \frac{\partial L}{\partial \theta_i} \qquad i=1,2,3 \tag{3.2}$$

where L is the Lagrangian function. It is the difference between the total kinetic energy K and the total potential energy P. The kinetic energy of each link is made up of the translation and rotation components. To find the kinetic energies of all the joints, both the linear and angular velocities have to be found. The potential energy for a planar manipulator is not taken into account since the manipulator is assumed to be in the horizontal plane and there are no gravitational effects.

The kinetic energies of all the joints are calculated from the kinematics of the manipulator. The translational kinetic energy of each link is found from the velocity at the mid-point of that link. Let (x_1,y_1), (x_2,y_2), and (x_3,y_3) denote the center points of the first, second and third link respectively as shown in Fig 3.1. The position coor-

dinates of the center of link 1 are

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} \frac{l_1}{2} c_1 \\ \frac{l_1}{2} s_1 \end{bmatrix} \tag{3.3}$$

The x and y components of their velocities are

$$\begin{bmatrix} \dot{x}_1 \\ \dot{y}_1 \end{bmatrix} = \begin{bmatrix} -\frac{l_1}{2} s_1 \dot{\theta}_1 \\ \frac{l_1}{2} c_1 \dot{\theta}_1 \end{bmatrix} \tag{3.4}$$

The resultant velocity is the square root of the sum of the square of the components and is given by

$$v_1 = \sqrt{\dot{x}_1^2 + \dot{y}_1^2}$$

$$= \frac{l_1}{2} \dot{\theta}_1 \tag{3.5}$$

There are more assumptions to be made before the kinetic energy of joint one can be found. The mass of each link is assumed to be evenly distributed along the link and the link is assumed to be a thin rod. These assumptions are made to facilitate the calculation of the moment of inertia of the links because it is extremely difficult to calculate the exact moment of inertia from the actual mass distribution of the links of the manipulator. The moment of inertia of a thin rod that rotates about its end point is known to be $I = \frac{1}{3} m l^2$, where m and l are the mass and length of the rod respectively.

The moment of inertia at the center of the link can be found by applying the parallel axis theorem to transfer the moment of inertia from the end of the link to the center of the link. The equivalent inertia is $\frac{1}{12}ml^2$, thus the rotational kinetic energy for joint i is expressed as

$$\frac{1}{2}(\frac{1}{12}m_i l_i^2)\dot{\theta}_i^2 \tag{3.6}$$

The total kinetic energy of joint one can now be found from expressions (3.5) and (3.6) to be

$$\frac{1}{6}m_1 l_1^2 \dot{\theta}_1^2 \tag{3.7}$$

The total kinetic energy of the other two links can be found in a similar way. Since there are no potential energy terms, the Lagrangian function is merely the sum of all the kinetic energy terms. The Lagrangian function after simplification can be expressed as follows

$$\begin{aligned} L = {} & \frac{1}{2}(a_3 + a_4 + a_5 c_2 + a_7 + a_6 c_{23} + a_8 c_3)\dot{\theta}_1^2 \\ & + \frac{1}{2}(a_7 + a_2 + a_8 c_3)\dot{\theta}_2^2 + \frac{1}{2}a_1 \dot{\theta}_3^2 \\ & + (a_7 + a_2 + \frac{1}{2}a_5 c_2 + \frac{1}{2}a_6 c_{23} + a_8 c_3)\dot{\theta}_1\dot{\theta}_2 \\ & + (a_1 + \frac{1}{2}a_6 c_{23} + \frac{1}{2}a_8 c_3)\dot{\theta}_1\dot{\theta}_3 + (a_1 + \frac{1}{2}a_8 c_3)\dot{\theta}_2\dot{\theta}_3 \end{aligned} \tag{3.8}$$

where all the a's are the constants that contain the masses and the link lengths of the manipulator. They are defined as

$$a_1 = \frac{1}{3} m_3 l_3^2 \tag{3.9.1}$$

$$a_2 = a_1 + \frac{1}{3} m_2 l_2^2 \tag{3.9.2}$$

$$a_3 = a_2 + \frac{1}{3} m_1 l_1^2 \tag{3.9.3}$$

$$a_4 = (m_2 + m_3) l_1^2 \tag{3.9.4}$$

$$a_5 = (m_2 + 2m_3) l_1 l_2 \tag{3.9.5}$$

$$a_6 = m_3 l_1 l_3 \tag{3.9.6}$$

$$a_7 = m_3 l_2^2 \tag{3.9.7}$$

$$a_8 = m_3 l_2 l_3 \tag{3.9.8}$$

With the Lagrangian function defined by (3.8) the joint torques can now be found from the Lagrangian equation (3.2). To find the first term of the torque of the first joint, the partial derivative of the Lagrangian function is taken with respect to the velocity of joint one and then the time derivative is taken. The second term of (3.2) can be found by taking the partial derivative of the Lagrangain with respect to the angle of joint one. The torques of the second and third joint are found in a similar way. Going through the above mentioned algebraic work and simplification for all the three joints, the joint torques are expressed as

$$\tau_1 = a_3 + a_4 + a_5 c_2 + a_7 + a_6 c_{23} + a_8 c_3 \ddot{\theta}_1^2$$

$$+ (a_7 + a_2 + \frac{1}{2} a_5 c_2 + \frac{1}{2} a_6 c_{23} + a_8 c_3) \ddot{\theta}_2$$

$$+ (a_1 + \frac{1}{2} a_6 c_{23} + \frac{1}{2} a_8 c_3) \ddot{\theta}_3$$

$$- (a_6 s_{23} + a_5 s_2) \dot{\theta}_1 \dot{\theta}_2 - (a_6 s_{23} + a_8 s_3) \dot{\theta}_1 \dot{\theta}_3$$

$$- (\frac{1}{2} a_5 s_2 + \frac{1}{2} a_6 s_{23}) \dot{\theta}_2^2 - (a_6 s_{23} + a_8 s_3) \dot{\theta}_2 \dot{\theta}_3$$

$$- (\frac{1}{2} a_6 s_{23} + \frac{1}{2} a_8 s_3) \dot{\theta}_3^2 \qquad (3.10.1)$$

$$\tau_2 = (a_7 + a_2 + \frac{1}{2} a_5 c_2 + \frac{1}{2} a_6 c_{23} + a_8 c_3) \ddot{\theta}_1$$

$$+ (a_7 + a_2 + a_8 c_3) \ddot{\theta}_2 + (a_1 + \frac{1}{2} a_8 c_3) \ddot{\theta}_3$$

$$+ (\frac{1}{2} a_5 s_2 + \frac{1}{2} a_6 s_{23}) \dot{\theta}_1^2$$

$$- a_8 s_3 \dot{\theta}_1 \dot{\theta}_3 - a_8 s_3 \dot{\theta}_2 \dot{\theta}_3 - \frac{1}{2} a_8 s_3 \dot{\theta}_3^2 \qquad (3.10.2)$$

$$\tau_3 = (a_1 + \frac{1}{2} a_6 c_{23} + \frac{1}{2} a_8 c_3) \ddot{\theta}_1 + (a_1 + \frac{1}{2} a_8 c_3) \ddot{\theta}_2 + a_1 \ddot{\theta}_3$$

$$+ (\frac{1}{2} a_6 s_{23} + \frac{1}{2} a_8 s_3) \dot{\theta}_1^2 + a_8 s_3 \dot{\theta}_1 \dot{\theta}_2 + \frac{1}{2} a_8 s_3 \dot{\theta}_3^2 \qquad (3.10.3)$$

From the above torque equations, the mass inertia matrix can be formed from the coefficients of $\ddot{\theta}_i$. The i^{th} row of the inertia matrix is composed of the coefficients of the torque equation of joint i and the j^{th} entry of each row corresponds to the coefficient of $\ddot{\theta}_j$. Collecting terms from the above torque equations, the inertia matrix can be written as

$$\begin{bmatrix} a_3+a_4+a_5c_2+a_7+a_6c_{23}+a_8c_3 & a_7+a_2+\frac{1}{2}(a_5c_2+a_6c_{23})+a_8c_3 & a_1+\frac{1}{2}(a_6c_{23}+a_8c_3) \\ a_7+a_2+\frac{1}{2}(a_5c_2+a_6c_{23})+a_8c_3 & a_7+a_2+a_8c_3 & a_1+\frac{1}{2}a_8c_3 \\ a_1+\frac{1}{2}(a_6c_{23}+a_8c_3) & a_1+\frac{1}{2}a_8c_3 & a_1 \end{bmatrix} \tag{3.11}$$

the Coriolis and centrifugal forces $V(\theta,\dot{\theta})$ can also be written as a vector from the joint velocities terms of the above torque equations

$$\begin{bmatrix} -(a_6s_{23}+a_5s_2)\dot{\theta}_1\dot{\theta}_2-(a_6s_{23}+a_8s_3)\dot{\theta}_1\dot{\theta}_3-\frac{1}{2}(a_5s_2+a_6s_{23})\dot{\theta}_2^2-(a_6s_{23}+a_8s_3)\dot{\theta}_2\dot{\theta}_3-\frac{1}{2}(a_6s_{23}+a_8s_3)\dot{\theta}_3^2 \\ \frac{1}{2}(a_5s_2+a_6s_{23})\dot{\theta}_1^2-a_8s_3\dot{\theta}_1\dot{\theta}_3-a_8s_3\dot{\theta}_2\dot{\theta}_3-\frac{1}{2}a_8s_3\dot{\theta}_3^2 \\ \frac{1}{2}(a_6s_{23}+a_8s_3)\dot{\theta}_1^2+a_8s_3\dot{\theta}_1\dot{\theta}_2+\frac{1}{2}a_8s_3\dot{\theta}_3^2 \end{bmatrix} \tag{3.12}$$

The above derived dynamic equation will be used in the following chapters. The numerical values used for simulation is summarized in the following table.

i	l_i (cm)	m_i (kg)	$\theta_{i__}$ (degree)
1	50.0	30.0	180
2	43.0	25.0	120
3	35.0	20.0	120

Table 3.1 Parameters of the three link redundant manipulator

Chapter 4

Time Optimal Control of A Redundant Manipulator

4.1 Introduction

Minimum time for non-redundant manipulators moving from one point to another point along a specified path is unique for there is only one unique set of joint trajectories that can satisfy the end effector requirement. However, for redundant manipulators there are infinitely many possibilities for one end effector position. Having a different starting position for the same end effector path will give different choosing a different inverse kinematic solution will immediately imply a different 'optimal time'. The minimum time algorithm developed here is therefore dependent upon the method used for solving the inverse kinematics. The pseudoinverse is used here to find the inverse kinematic solution simply because there is no other known alternative that has been proved to have a lesser time than the popular pseudoinverse solution.

A time optimal control algorithm for a planar redundant manipulator is presented in this chapter. The end effector is to move from one point to another following a specified trajectory in a two dimensional plane given the configuration of the manipulator and the joint velocities at the starting point. The objective of the algorithm is to find a set of joint torques which will drive the manipulator from its given initial position to the final position along a specified desired path in minimum time.

The time optimal control for redundant manipulator is not attempted with the conventional optimal control method, the Pontryagin's maximum principle, for various reasons. It is because the conventional method is faced with a dimensionality problem. Even for a second order system, which is the case here, each extra degree of freedom for the redundant case implies two more states and co-states equations. Thus for the case considered here the dimension is six. The increase in system equations effectively reduces the chance of getting a convergent solution for a two point boundary value problem (TPBVP). It was pointed out that the solution does not converge for even the simple case of a three degree of freedom elbow-type manipulator [Bobrow et el-83]. Moreover, it would be rather difficult, if not impossible, to add the path as an additional constraint. The minimum time control approach derived by Bobrow for the non-redundant manipulator will be used instead for the redundant case. The major advantage of this algorithm is that an initial value problem is solved instead of the difficult TPBVP.

Bobrow's method for the time-optimal control of a non-redundant manipulator is described first. The vector that contains the displacement and velocity along the specified trajectory is the state vector. The non-linear dynamics and other constraints are transformed into state dependent constraints on the acceleration along the path. The acceleration along the path is actually the control variable. Time optimal solution is obtained from the acceleration profile which gives the maximum velocity along the path. Therefore, the joint angles and velocities are written as functions of the displacement. A second order differential equation can be formed with the displacement and velocity as state variables. This differential equation is integrated as an initial value problem in both the forward and backward directions to obtain the intersection of trajectories and hence the switching time.

The key to the extension of this method to the redundant case is that pseudoinverse is employed to obtain the joint angles as a function of the Cartesian position. This is made possible by taking the Cartesian velocities as the increment and using the pseudoinverse to find the corresponding joint increment. The solution to the problem is divided into two main categories, the single switching case and the multiple switching case. Multiple switchings occur when the available torque is not able to meet the velocity requirements.

4.2 Time Optimal Control of Non-Redundant Manipulators

A simple planar two link non-redundant manipulator (Figure 4.1) is chosen for the derivation of the algorithm. The manipulator is assumed to have rigid links and friction is not taken into account. Lagrange's equations will be used to describe the manipulator dynamics. The dynamics of the end effector are assumed to have negligible effect on the manipulator dynamics. The orientation is assumed to be unchanged.

The end effector is required to move along the specified path from $\mathbf{P}_0$ to $\mathbf{P}_f$. The initial and final velocities are taken as zero. The dynamics of the manipulator based on the Lagrangian approach have the following form

$$M(\underline{\theta})\ddot{\underline{\theta}} + h(\underline{\theta}, \dot{\underline{\theta}}) = \mathbf{T} \qquad (4.1)$$

where $\underline{\theta}$ and $\mathbf{T}$ are vectors of the joint angles and joint torques respectively. $M(\underline{\theta})$ is a square matrix of the mass moment of inertia and $h(\underline{\theta},\dot{\underline{\theta}})$ is the vector of the centrifugal and Coriolis terms.

The constraints on the actuator torques are expressed as arbitrary functions of the joint angles and velocities

$$T_{i_}(\underline{\theta}, \dot{\underline{\theta}}) \leq T_i \leq T_{i_}(\underline{\theta}, \dot{\underline{\theta}}). \qquad (4.2)$$

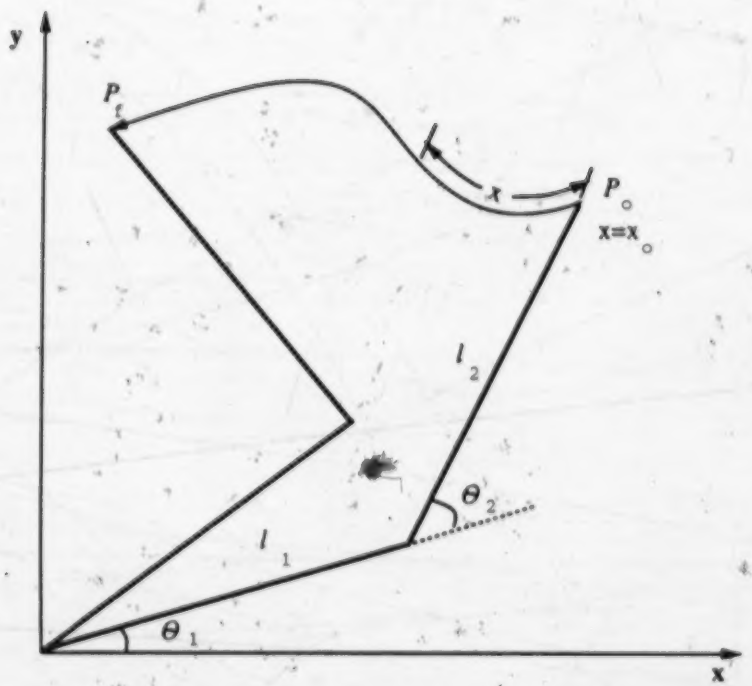

Figure 4.1 A two link planar manipulator

The objective is to find the joint torques $\mathbf{T}(t)$ which drive the end effector from the initial position $\mathbf{P}_0$ to the final position $\mathbf{P}_f$ in a minimum time, along the specified path.

Let x be the distance traveled by the end effector measured from $\mathbf{P}_0$ and let x_0 and x_f be the initial and final distance moved by the tip of the arm, with path velocity and acceleration of the tip of the manipulator denoted by $\dot{x}$ and $\ddot{x}$ respectively. The key to solving this time optimal problem is to transform the system to an equivalent system with a single control variable, $\ddot{x}$ the acceleration along the trajectory. This implies that the joint angles have to be computed as a function of x and the angular velocities and accelerations as functions of x, $\dot{x}$ and $\ddot{x}$, that is

$$\theta = \theta(x) \tag{4.3.1}$$

$$\dot{\theta} = \theta(x, \dot{x}) \tag{4.3.2}$$

$$\ddot{\theta} = \theta(x, \dot{x}, \ddot{x}) \tag{4.3.3}$$

The above equations may look simple but it is extremely difficult or almost impossible to express them analytically even for the planar manipulator considered here. They depend upon the geometry of the specified path and also the inverse kinematics of the manipulator. For these reasons equations 4.3 will be computed numerically for each x.

The difficulty of expressing the joint angles, velocities and accelerations in terms of the trajectory displacement can be reduced by decomposing the problem into two parts, the evaluation of the inverse kinematics of the manipulator and the expression of the Cartesian co-ordinates of a particular path in terms of the path parameter x. The latter part is no more than a particular way of defining the trajectory. That is,

$$\mathbf{r} = Traj(x) \tag{4.4.1}$$

The trajectory is to be specified as function *Traj* with the distance traveled by the end effector, x, as the independent variable and the Cartesian position of the manipulator $\mathbf{r}$ as the output. For example, if the end effector is to move horizontally from right to left, starting at (x_0, y_0), then

$$\begin{bmatrix} r_x \\ r_y \end{bmatrix} = \begin{bmatrix} x_0 - x \\ y_0 \end{bmatrix} \tag{4.4.2}$$

Now, recall the kinematic equation

$$\mathbf{r} = r(\underline{\theta}) \tag{4.5.1}$$

If we denote r^{-1} as the inverse kinematic function of the manipulator, the inverse kinematic equation can be written as

$$\underline{\theta} = r^{-1}(\mathbf{r}) \tag{4.5.2}$$

There are generally a few options to solve the inverse kinematics. For manipulators with simple kinematics, the closed form analytic solution is used; but for more involved kinematics, either a table look up or numerical iteration is used. The inverse function of the joint angle can now be obtained by substituting (4.4.1) into (4.5.2) to give

$$\theta(x) = r^{-1}(Traj(x)) \tag{4.5.3}$$

Thus, the inverse function can be obtained as long as the inverse kinematic function is defined at each point of the trajectory. In order to take the time derivative of (4.5.3), its right hand side must be piecewise continuous and differentiable for all x. This indirectly imposes the condition that the specified path be a smooth function. The derivative of equation (4.5.3) is not used for the evaluation of $\dot{\theta}$ for it is easier to take

the derivative of (4.4.1). The change of the position vector **r** with respect to time is equivalent to the scalar change of x with respect to time, $\dot{x}$ multiplied by the unit vector which is tangential along the trajectory. Let this unit vector be **v**, the derivative of (4.4.1) is written as

$$\frac{d}{dt}\mathbf{r} = \dot{x}\,\mathbf{v} \tag{4.6.1}$$

Differentiate the kinematic equation (4.5.1) with respect to time and substitute (4.6.1) on the left side get

$$\dot{x}\,\mathbf{v} = J(\theta)\dot{\theta} \tag{4.6.2}$$

where $J(\theta)$ is the Jacobian matrix of the partial derivatives of the joint angles. When the Jacobian matrix is not singular, the joint velocities can be expressed as

$$\dot{\theta}(x,\dot{x}) = J^{-1}(\theta)\dot{x}\,\mathbf{v} \tag{4.6.3}$$

Singular configurations of the manipulator can easily be avoided in the path planning stage, therefore the Jacobian can always be made invertible. Again, (4.6.3) is not used to find the joint accelerations, instead equation (4.6.2) is differentiated as

$$\mathbf{v}\ddot{x} + \mathbf{w}\dot{x}^2 = J(\theta)\ddot{\theta} + \dot{J}(\theta)\dot{\theta} \tag{4.7.1}$$

where **w** is the derivative of **v** with respect to x and $\dot{J}(\theta)$ is the derivative of the Jacobian matrix with respect to time. It is noted that the evaluation is more complicated than it looks. The derivative of the unit vector **v** with respect to **x** is best understood as the incremental change of the tangential vector or tangential acceleration. Since there is no increment in magnitude for a unit vector, **w** can be considered as another vector normal to the unit vector **v**. The second term on the left side of (4.7.1) is there-

fore the normal acceleration of the tip of the manipulator. The complexity in solving (4.7.1) lies in taking the time derivatives of the Jacobian matrix. Suppose a_{ij} is the element on the i^{th} row and j^{th} column of the Jacobian matrix, the corresponding time derivative is expressed as

$$\frac{da_{ij}}{dt}=\sum_{k=1}^{n}\frac{\partial a_{ij}}{\partial \theta_k}\dot{\theta}_k \tag{4.7.2}$$

where n=2 for the case considered here. The joint accelerations can be expressed explicitly after rearranging terms of (4.7.1) as

$$\ddot{\theta}(x,\dot{x},\ddot{x})=J^{-1}(\theta)(\,\mathbf{v}\ddot{x}+\mathbf{w}\dot{x}^2-\dot{J}(\theta)\dot{\theta}\,) \tag{4.7.3}$$

The purpose of expressing the joint acceleration $\ddot{\theta}$ as a function of x, $\dot{x}$ and $\ddot{x}$ is, after all, to find expressions for the maximum accelerations and decelerations of the end effector along the trajectory. Substituting equation (4.7.3) into the dynamic equation (4.1), the torque vector T can be written explicitly as functions of x, $\dot{x}$ and $\ddot{x}$

$$T=\ddot{x}C_1(x)+C_2(x,\dot{x}) \tag{4.8.1}$$

where $C_1(x)$ and $C_2(x,\dot{x})$ are vectors defined as follows

$$C_1(x)=M(\theta)J(\theta)^{-1}\mathbf{v} \tag{4.8.2}$$

$$C_2(x,\dot{x})=M(\theta)J(\theta)^{-1}(\mathbf{w}\dot{x}^2-\dot{J}(\theta)\dot{\theta}\,)+h(\theta,\dot{\theta}) \tag{4.8.3}$$

The torque of the i^{th} joint in the vector equation (4.8.1) must satisfy the torque constraint of (4.2). Therefore, substitute it into (4.2) and rearrange terms so that the bounds of the trajectory acceleration are explicitly expressed in terms of the torque constraints of each joint. The dependence of T_{max} and T_{min} on θ and $\dot{\theta}$ can be

replaced by dependence on x and $\dot{x}$ through equations (4.3.1) and (4.3.2). The constraints on the acceleration $\ddot{x}$ is given by

$$f_i(x,\dot{x}) \le \ddot{x} \le g_i(x,\dot{x}), \qquad i=1,2 \tag{4.9.1}$$

where f and g are defined by

$$f_i(x,\dot{x}) = \begin{cases} \dfrac{(T_{i_{max}} - C_{2i})}{C_{1i}}, & C_{1i}>0 \\ \dfrac{(T_{i_{min}} - C_{2i})}{C_{1i}}, & C_{1i}<0 \end{cases} \tag{4.9.2}$$

$$g_i(x,\dot{x}) = \begin{cases} \dfrac{(T_{i_{min}} - C_{2i})}{C_{1i}}, & C_{1i}>0 \\ \dfrac{(T_{i_{max}} - C_{2i})}{C_{1i}}, & C_{1i}<0 \end{cases} \tag{4.9.3}$$

Equations (4.9.2) and (4.9.3) are valid only when C_{1i} is non zero. They give the range of the end effector acceleration $\ddot{x}$ that the actuator motors are capable to produce. When condition (4.9.1) is violated, the joint torques will not be sufficient to keep the tip in the desired path. This can happen when the range of $\ddot{x}$ in (4.9.1) for the individual joints does not overlap. That is when the trajectory acceleration requirement for the joints cannot be all satisfied at the same time. In that case, the trajectory velocity is too large for the joint torques to realize and the tip will start to deviate from the specified path. Therefore, the necessary and sufficient conditions for the end effector to stay in the path is that the intervals $[f_i(x,\dot{x}), g_i(x,\dot{x})]$ for all the joints must have a nonempty intersection and that $C_{1i}(x,\dot{x}) \neq 0$ for all i. From equation (4.8.1), it

is seen that $\ddot{x}$ is arbitrary for joint i if C_{1i} is zero and $\ddot{x}$ can be determined from the other constraints in (4.8.1). The remaining concern is the possible situation where C_{1i}'s are zero for all the joints. This will not occur if the manipulator is not at a singular position. The mass matrix M is always positive definite and v is a unit vector, it follows from (4.8.2) that C_1 will not be a null vector if the Jacobian is invertible. Any trajectory acceleration $\ddot{x}$ which satisfy (4.9) for a given x and $\dot{x}$ is known as an admissible acceleration. From the above discussion, the admissible acceleration can now be redefined as

$$f(x,\dot{x}) \le \ddot{x} \le g(x,\dot{x}) \tag{4.10.1}$$

where

$$f(x,\dot{x}) = \max_i f_i(x,\dot{x}) \tag{4.10.2}$$

$$g(x,\dot{x}) = \min_i g_i(x,\dot{x}) \tag{4.10.3}$$

The maximum and minimum of f_i and g_i are taken over those with non-zero C_{1i}. For when any two of the intervals of $[f_i(x,\dot{x}), g_i(x,\dot{x})]$ do not overlap, $f_i(x,\dot{x}) > g_i(x,\dot{x})$ and equation (4.10.1) cannot be satisfied. Having an admissible acceleration or $f_i(x,\dot{x}) \le g_i(x,\dot{x})$ is crucial in solving this time optimal problem. The time optimal control problem can now be restated mathematically as follows: determine $\ddot{x}(t)$, subject to 4.10.1, such that t_f, the final time is minimized with given initial and final trajectory displacement and velocity. The acceleration along the trajectory is now the control variable and the torque constraints are embedded into equation (4.10.1).

4.3 Time Optimal Control of Redundant Manipulator-Problem Formulation

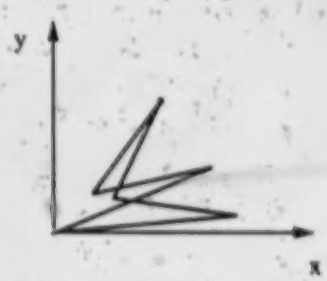

Figure 4.2 Two possible configurations for a 3-link planar redundant manipulator

For non-redundant manipulators, there are only a few possible postures for any Cartesian position and it is easy to specify the inverse kinematics as a function. However, the redundant manipulator kinematics is not unique and there are simply infinitely many possibilities for any point inside the workspace. Figure 4.2. shows two possible configurations of a three link manipulator for the same Cartesian co-ordinates. Unlike the non-redundant case, a closed form inverse kinematics does not exist for the redundant case. The redundant kinematics have more unknowns than the number of equations. It is difficult to obtain a unique solution by iteration. The major difficulty in the application of the above control algorithm to the redundant manipulator comes from the fundamental requirement that the joint angles be a function of the Cartesian co-ordinates. The nature of this algorithm only allows one set of uniquely defined joint angles for each Cartesian position. Obviously, inverse kinematics of the redundant manipulator must be defined somehow.

The trick here is to use the pseudoinverse to look for a unique solution. The pseudoinverse is particularly suitable for this application because of its generic properties of uniqueness and minimum norm. It has been widely used to resolve the

kinematic problem for redundant manipulators. However, the problem here is that it is only applicable for resolution of redundancy at the velocity level or acceleration level. To get around this problem, the Cartesian velocities in the x and y direction are taken as increments to the x and y Cartesian position. The pseudoinverse of the Jacobian can then be used to find the corresponding increment in the joint angles. Instead of solving the inverse kinematics directly, the joint angles are found according to the previous joint angles. This implicitly requires an additional set of starting angles to be specified for the non-redundant case. This is due to the infinitely many possible joint configurations for any given Cartesian and starting at different postures means totally different intermediate joint trajectories. Therefore, at time t, the joint angles for the next time interval can be expressed as

$$\theta(x(t+dt)) = \theta(x(t)) + d\theta \qquad (4.11.1)$$

where $x(t)$ is the trajectory displacement at t, $x(t+dt) = x+dx$, and dx is the change in x during the time interval. The corresponding increment in θ is denoted as $d\theta$ and is evaluated from

$$d\theta = J^{+}(\theta(x))\, d\mathbf{r} \qquad (4.11.2)$$

in which $J^{+}(\theta(x))$ is the pseudoinverse of the Jacobian evaluated at $\theta(x)$ and $d\mathbf{r}$ is the incremental change of the path in the Cartesian co-ordinates. The incremental change $d\mathbf{r}$ is actually evaluated from the difference of $\mathbf{r}$ in equation (4.4.1) at time t and $t+dt$, it can be written as

$$d\mathbf{r} = Traj(x(t+dt)) - Traj(x(t)) \qquad (4.11.3)$$

Now, the inverse kinematics is written as a function of x, the former displacement, and dx, the relative change in x compared with the previous time interval.

Substituting (4.11.2) and (4.11.3) into (4.11.1), θ can be written as

$$\theta(x + dx) = \theta(x) + J^{+}(\theta(x))[\, Traj(x+dx) - Traj(x) \,] \qquad (4.11.4)$$

The inverse kinematic solution obtained from the above formulation is formed to be satisfactory; therefore it is not necessary to use a more sophisticated method to evaluate (4.11.2). From (4.11.2), it is seen that $d\mathbf{r}$ is always a small quantity and the pseudoinverse is the only cause for any misbehavior. The pseudoinverse provides proper solutions unless the Jacobian matrix is rank deficient. Again, singularity is assumed to be avoided in the path planning stage, therefore the Jacobian is always at its full rank.

Similarly, the joint velocities are not found by taking the time derivative of the equation (4.11.4), rather equation (4.6.3) is used. The derivation of (4.6.3) is the same as that in the non-redundant case, except that the Jacobian matrix is non-square in the redundant case and the pseudoinverse is used. The use of the pseudoinverse at the velocity level is justified because the inverse kinematic solution is actually obtained by the pseudoinverse as well. The joint velocities found by the pseudoinverse is equivalent to taking the time derivative of (4.11.4). Replacing the Jacobian inverse by the pseudoinverse in the above derivation from (4.6.3) onwards, the algorithm for the redundant case follows exactly as for the non-redundant case.

4.4 Solution to the Time Optimal Problem for the Redundant Case

With the above formulation, the minimum time can now be found by choosing from the admissible accelerations governed by (4.10.1) a unique $\ddot{x}(t)$ that will make the velocity $\dot{x}$ as large as possible at every point of $x(t)$. The idea behind this approach is similar to the bang bang control strategy in the classical optimal control theory [Athans & Falb 66]. Minimum time can be achieved by first driving the

manipulator at the maximum allowable trajectory acceleration and then switching to maximum deceleration at an appropriate switching time t_s. This will be proved later in this chapter. The solution to the problem of obtaining the control law, after all, is to find the single or multiple switching times t_s's that will bring the end effector from the initial boundary condition to the final boundary condition in minimum time subject to the previously mentioned assumptions and conditions.

4.4.1 Single Switching Case

The solution to this time optimal problem is best explained through a switching curve in the x—$\dot{x}$ phase plane [Bobrow 83]. The construction of the phase plane trajectory for the single switching case is rather straight forward. A typical single switching minimum time phase plane trajectory is shown in Figure 4.3. The end effector starts to accelerate at $\ddot{x}=g(x,\dot{x})$ from x_0 to x_s and decelerate at $f(x,\dot{x})$ from x_s to x_f, x_s is the distance moved by the end effector at the switching time t_s.

In Figure 4.3, the maximum acceleration phase trajectory is obtained by integrating $\ddot{x}=g(x,\dot{x})$ for x, and $\dot{x}$, for increasing t from x_0 to some point a shown in the Figure 4.3. Then, the maximum deceleration phase trajectory $\ddot{x}=f(x,\dot{x})$ is found by integrating backward in time from x_f to until it intersects with the forward trajectory at x_s. The minimum time can actually be found for the manipulator with any starting and finishing point on the phase plane as long as the acceleration required is within allowable limits.

Assume that the time required to move the tip from the initial position x_0 to the final position x_f is finite and is given by

$$t_f = \int_{x_0}^{x_f} \frac{dx}{\dot{x}} \qquad (4.12.1)$$

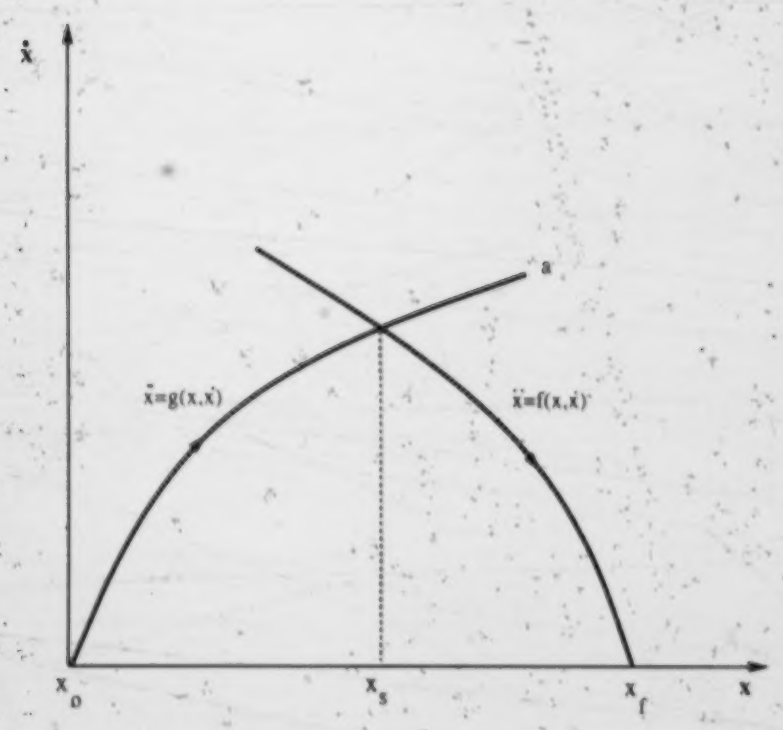

Figure 4.3 Time optimal trajectory for the single switching case

Suppose t_f is the minimum time, then $\dot{x}$ in the above equation has the maximum velocity profile. $\dot{x}$ is obtained by integrating the following differential equation

$$\ddot{x}(\dot{x},x) = \begin{cases} g(x,\dot{x}) & t < t_s \\ \\ f(x,\dot{x}) & t > t_s \end{cases} \qquad (4.12.2)$$

In the phase plane in Figure 4.3, with the same starting point, all the possible phase plane trajectories lie between the extreme solutions for maximum acceleration and maximum deceleration, with the maximum acceleration solution always having a higher velocity. Suppose $\hat{x}$ is another solution that gives shorter time, it implies that it possesses a higher velocity than $\dot{x}$ in Figure 4.3 at some point between x_0 and x_f. Since the phase plane trajectory from x_0 to x_s already has maximum velocity, it is not possible for $\hat{x}$ to be larger than $\dot{x}$. Any velocity higher than the minimum velocity solution from the maximum deceleration cannot satisfy the final boundary condition.

The solution of the single switching case actually involves solving two initial value differential equations, one in the forward direction and the other backward. This implies that both the joint angles at the beginning and at the end of the trajectory are known. The difficulty here for the redundant case is that there are infinitely many possibilities for the joint angles at the end point. A different set of initial joint angles for the backward integration obviously gives a different phase plane trajectory. Intersection in the phase plane can still be found, however, the switching time obtained is incapable of producing the desired trajectory. The reason being is at that particular point of intersection, the joint angles inherited from the forward integration do not correspond with those inherited from the backward integration even though both of them have the same Cartesian co-ordinate. This implies that there is no choice for the initial conditions for the backward integration from the end point. Solution to this

problem lies in the uniqueness of the inverse kinematics discussed earlier. The end point 'initial' conditions can actually be obtained from the forward integration from the initial point x_0 to the final point x_f considering only the kinematics. The joint angles at x_f are then used as the initial conditions for the backward integration. Since the pseudoinverse solution to the inverse kinematics is unique, the joint angles at the phase plane intersection are guaranteed to have the same joint angles as in the non-redundant case.

4.5 Computational Problems

In this section, some techniques to streamline the computation are discussed.

4.5.1 Integration Technique

Among all the various algorithms suggested in the literature for solving a system of initial boundary value differential equations, it is almost impossible to find one single method that has best performance in all measurable parameters. There is always a compromise between computation time and numerical accuracy. The most important factor, however, is not the performance of the integration method but the behavior of the differential system. Prior knowledge of the system is extremely important for the choice of a suitable integration method because there are methods which are designed to handle only specific behaviors of the system. Stiffness is maybe a typical special behavior that requires special treatment. Because of the complexity of the dynamic equations, it is difficult to prove that the system is stiff or otherwise. However, from the restriction of a rapid change of joint angles and joint velocities, it is unlikely that the dynamics is stiff.

The major difficulty in choosing a suitable integration algorithm does not lie in any abnormal behavior of the dynamic equations, but on the dependence of these equa-

tions upon the input torques. The system behaves according to the behavior of the input torques. Discontinuity in the joint torques is almost guaranteed for the bang-bang type optimal control and this imposes restrictions in the selection of an appropriate integration method.

The numerical integration algorithms can be classified into three categories, the approximation of the actual solution by linear combination of independent functions, the approximation by the first few terms of an orthogonal function and the approximation at some specified points in the integration interval. The last method is also known as the discrete variable method and is more universally applicable than the other two. There are two distinct conventional discrete variable methods of integration, the one-step or Runge-Kutta (RK) method and the Linear Multistep method (LMM). As seen from the derivation of these algorithms [Lambert 73], the differential equations are assumed to be continuously differentiable in the entire domain of the integration. These methods are therefore not suitable for the simulation of the time optimal control. However, a combination of the low order discretization scheme and the extrapolation method can handle discontinuities and even singularities of a differential equation.

4.5.2 Extrapolation Methods

Suppose a number A_0 is to be evaluated, and only the approximation $A(h)$ can be computed, where h is the discretization parameter and $A(h)$ approaches A_0 as $h \to 0$. That is $A(h)$ is assumed to have an asymptotic expansion of the following form for every fixed N

$$A(h) = A_0 + \sum_{i=1}^{N} A_i h^N + R_N(h) \qquad (4.13.1)$$

where $A_1, A_2, \cdots$ are coefficients independent of h and $R_N(h)$, the error term, is of order $O(h^{N+1})$ as $h \to 0$. For $h = h_0$, $A(h_0)$ and $A(\frac{1}{2}h_0)$ can be calculated from equation (4.13.1) and when $h \to 0$ both $A(h_0)$ and $A(\frac{1}{2}h_0)$ approach $A_0 + O(h_0)$. However, a linear combination of $A(h_0)$ and $A(\frac{1}{2}h_0)$ such as

$$2A(\frac{1}{2}h_0) - A_0 = A_0 + \frac{1}{2}A_2 h_0^2 + \cdots = A_0 + O(h_0^2) \qquad (4.13.2)$$

can yield a better approximation of A_0 than either $A(h_0)$ or $A(\frac{1}{2}h_0)$ as seen from the $O(h_0^2)$ term. Moreover, if $A(\frac{1}{4}h_0)$ is also used, a even better approximation which differs A_0 by $O(h_0^3)$ can be found. This is referred to as the basic idea of Richardson Extrapolation method. Now, consider a sequence $h_0, h_1, h_2, \cdots h_S$ of h values where $h_0 > h_1 > h_2 \cdots > h_S > 0$, a linear combination of $A(h_s)$ can be used to approximate A_0 with an error of $O(h_0^{S+1})$. The formation of such a linear combination is the same as doing a polynomial interpolation of $A(h_s)$ which is actually considered as a polynomial with h_s as an independent variable. Instead of doing interpolation, extrapolation of h_s to zero is done to get the best approximation of A_0. An extrapolation table of order M can be formed as follow

$$\begin{array}{lllll}
h_0 & T_{00} & & & \\
h_1 & T_{10} & T_{01} & & \\
h_2 & T_{20} & T_{11} & T_{02} & \\
\cdot & \cdot & \cdot & \cdot & \\
\cdot & \cdot & \cdot & \cdot & \\
h_{M-1} & T_{M-1,0} & T_{M-2,0} & \cdots & T_{0,M-1} \\
h_M & T_{M,0} & T_{M-1,0} & \cdots & T_{1,M-1} \quad T_{0,M}
\end{array} \qquad (4.13.3)$$

where M is the order of extrapolation and $T_{0,M}$ is the h=0 extrapolation of order M. The second column of the above table is known as the zero-th column of the extrapolation table. Each entry T_{i0} in the zero-th column is actually the evaluation of $A(h_i)$. All the T's values in the subsequent columns are calculated iteratively by an extrapolation formula based upon the zero-th column. The extrapolation formula depends on the method of extrapolation used. Polynomial and rational function extrapolations are the two major types of extrapolation methods currently used. The rational function, developed by Bulirsch and Steor [Bulirsch and Steör 1966], is used here for it can handle singularity and discontinuity much better than the polynomial function, of course at the cost of more computation time. The extrapolation formula is a recursive formula that generates the T's of the non-zero column either by rows from top to bottom or by columns from left to right after the zero-th column has been evaluated. The rational function extrapolation formula can be expressed as [Fatunla 1988]

$$T_{r,0} = T_{r+1,s-1} + \frac{T_{r+1,s-1} - T_{r,s-1}}{(\frac{h_r}{h_{r+s}})^{\gamma} \left[1 - \frac{T_{r+1,s-1} - T_{r,s-1}}{T_{r+1,s-1} - T_{r+1,s-2}}\right] - 1} \qquad (4.13.4)$$

where s=1,2,...M and r=0,1,...M. It is seen from the above expression that evaluation of the first column requires $T_{r,-1}$ to be known and it is set to zero. γ is a constant depending upon the asymptotic error expansion in h. It is taken to be one when the error expansion is h and two when it is h^2. The choice of γ therefore depends upon the error expansion of the single step method used for the generation of the zero-th column. With the above equation and the zero-th column known, the M^{th} order extrapolation of $A(h_0)$ can be found.

The key to the use of this extrapolation method to solve a differential system lies in the generation of the zero-th column. Single step methods such as the Euler's rule, trapezoidal rule, mid-point rule and the Inverse Euler rule can all be used to generate the zero-th column. However, the Inverse Euler rule is known to perform best near a singularity. The Inverse Euler rule is actually adapted from the Inverse Polynomial method [Fatunla 1982]. Retaining only the first two terms of the denominator of the inverse polynomial, a one step integration formula is formed as follows:

$$y_{n+1} = \frac{y_n^2}{y_n - hy_n'} \tag{4.13.5}$$

This is easily applicable to a single differential equation, but for a system of equations, the division of matrices is not allowed. However, it has been proved [Lambert 73] that the Inverse Euler equation is component applicable to systems. That is, the above equation has to be applied individually to every single equation of the system.

4.5.3 Generation of the Zero-th Column

Before the evaluation of the zero-th column, the relation between successive meshsizes h_r and h_{r+1} has to be determined. There is no straight rule to define the h_r as long as h_{r+1} is smaller than h_r. The following relationship between the meshsizes is used

$$h_r = \frac{h}{2^{r+1}}, \qquad r=0,1,2,...M \tag{4.13.6}$$

Now, a sequence of integers is needed to generate the increment of h_r. The integers are defined as

$$N_r = \frac{h}{h_r}, \qquad r=0,1,2,...M \tag{4.13.7}$$

Assuming that $\mathbf{x}_n$ and $\mathbf{y}_n$ are the input and output vectors given at the $(n+1)^{th}$ step, a sequence of mesh points at which the differential equation is evaluated within the step can be defined as

$$t_s = x_n + sh_r, \qquad s=0,1,2,...N_r \tag{4.13.8}$$

It is noted that $t_0=x_n$ and $t_{N_r}=x_{n+1}$ correspond to the current and subsequent step of x. With the above expressions for h_r, N_r and t_s, the zero column can be generated using the Inverse Euler rule as

$$z_{s+1}^i = \frac{(z_s^i)^2}{z_s^i - h_r(z_s^i)'}, \qquad s=0,1,2,...N_r-1, \quad i=1,2,...,m \tag{4.13.9}$$

where m is the size of the system or the total number of differential equations. The superscript i denotes the i^{th} element of the vector z_s. Equation (4.13.9) is used repeatedly to find all the z_s^i with $z_0^i = y_n^i$ for all the elements in the vector. The term $(z_s^i)'$ is the evaluation of the differential equation at t_s, that is $(z_s^i)' = f_i(t_s, z_s)$. Finally, the elements of the zero-th column of the extrapolation table are defined as

$$T_{r0}^i = z_{N_r}^i = y^i(x_{n+1}, h_r) \tag{4.13.10}$$

The above formulation of a rational function extrapolation with the Inverse Euler method is used as an integration tool to solve the differential equation (4.12.2).

4.6 Examples

The time optimal control algorithm for the redundant manipulator discussed earlier is implemented on the three link manipulator model described in chapter 3. Two examples that require single switching are presented here. The maximum allowable torque vector is chosen such that the joint velocities and accelerations are reasonable. A larger maximum allowable torque is used for joint one and a smaller one is used for the last joint. This is done because the first joint usually needs a larger torque to produce the same angular motion than the other joints. Therefore, the maximum allowable joint torque vector is chosen to be (50.0,30.0,10.0) Newton meters. Variation of the torque vector directly influences the time required for the desired trajectory and the magnitude of the maximum velocity curve.

Suppose the end effector is to follow a straight line path that moves in the negative y direction starting at P_0(1.0, 0.6), the time optimal solution can be found with one switching. The path length is 0.5 meters and the end point P_f is therefore (1.0, 0.1). The initial joint angles are given as (0.12,0.406,0.659) in radians and the kinematics is evaluated in the forward direction to obtain the joint angles at the destination which is found to be (–0.519, 0.58, 1.109). The motion of the end effector is shown in Fig 4.4 which is obtained by drawing the position of the manipulator from the initial time to the final time in twenty equal time intervals. From the spacing between the links in Fig 4.4, it is noted that the end effector is slower at the beginning and at the end of the path. The maximum velocity curve is found by equating the forward and backward phase plane trajectories $f(x, \dot{x})=g(x, \dot{x})$ and solve for x and $\dot{x}$.

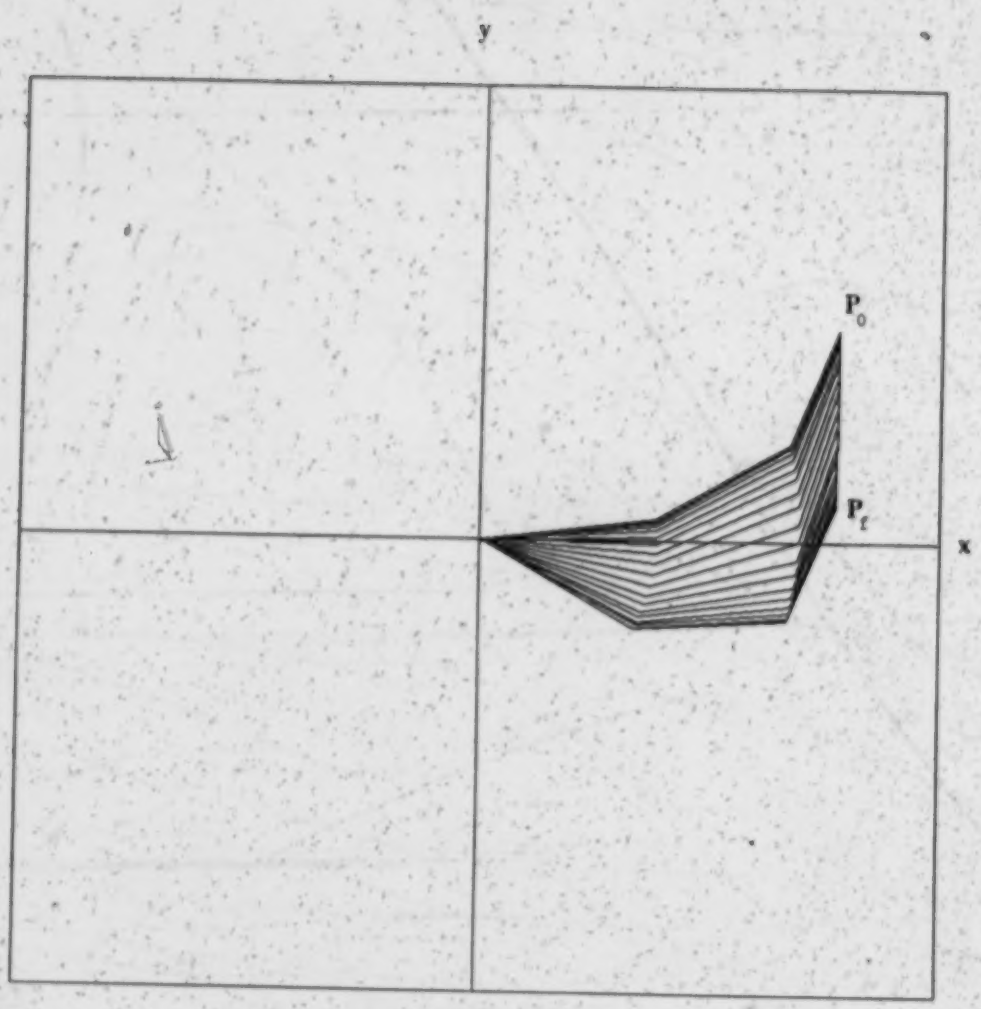

Figure 4.4 End effector motion

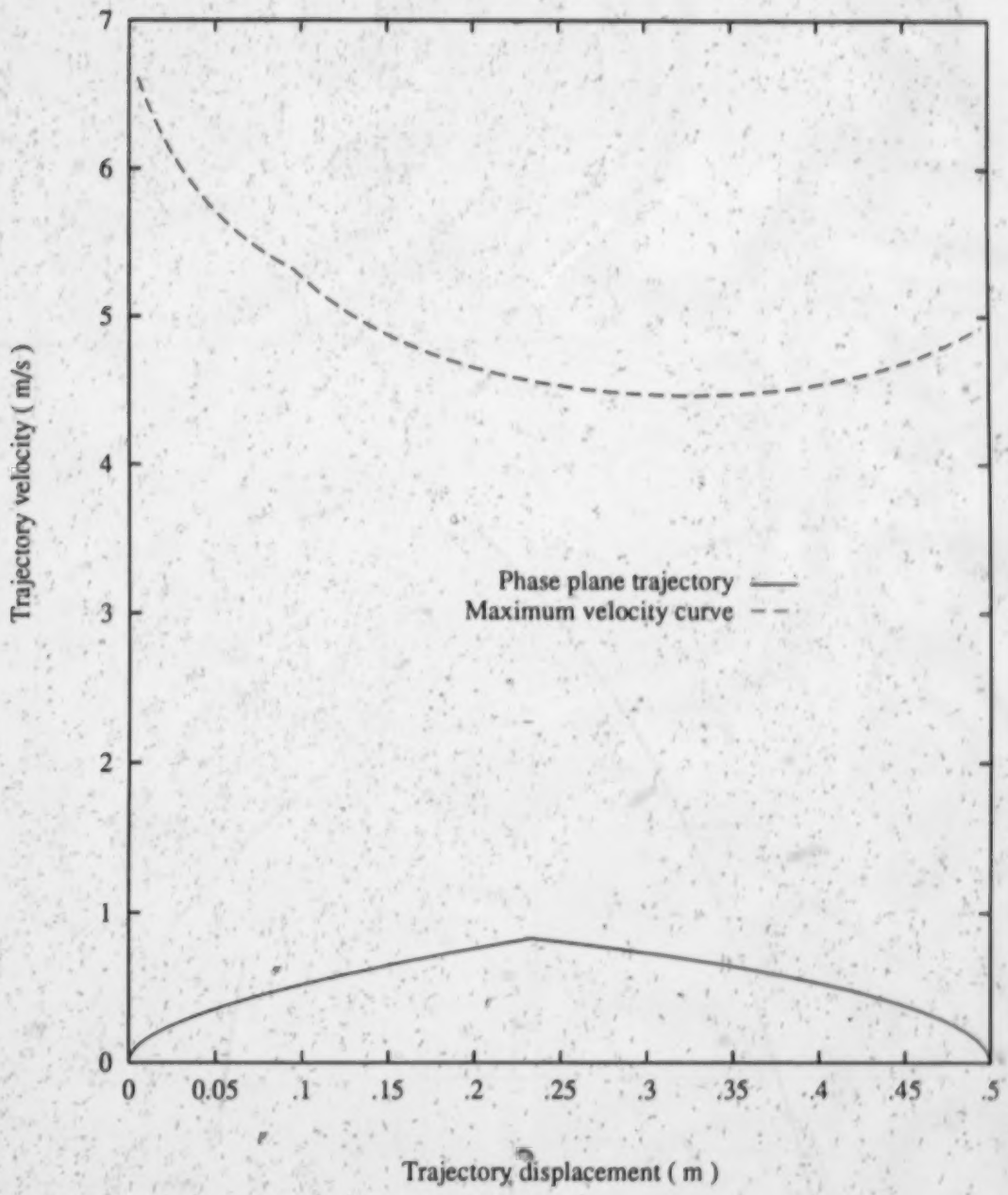

Figure 4.5 Phase plane trajectory and maximum velocity curve

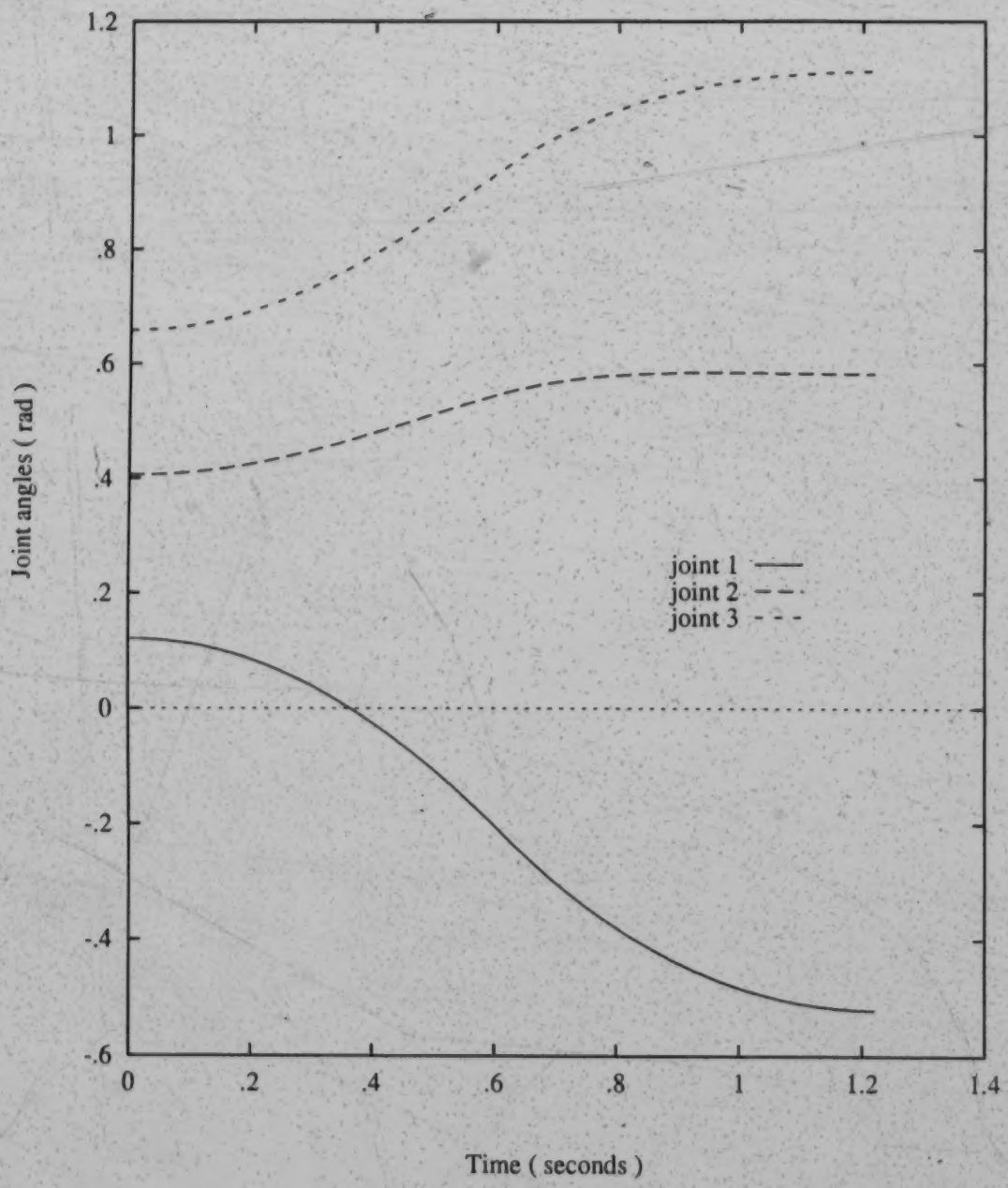

Figure 4.6 The resulting a) Joint angles, b)Joint velocities and c) Joint torques for the end effector motion

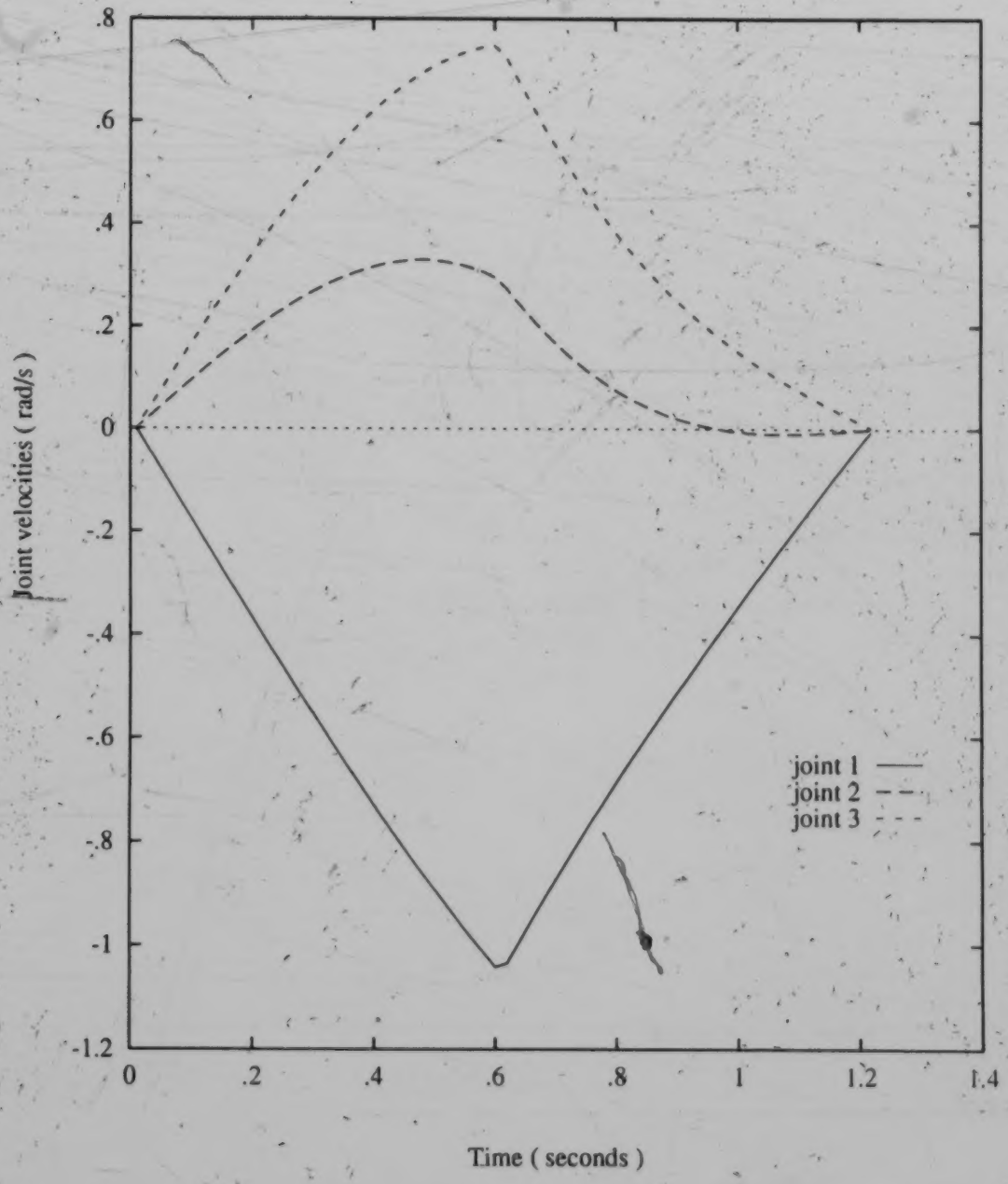
.8
.6
.4
.2
0
-.2
-.4
-.6
-.8
-1
-1.2
0
.2
.4
.6
.8
1
1.2
1.4
Joint velocities (rad/s)
Time (seconds)
joint 1
joint 2
joint 3

Figure 4.6 (b)

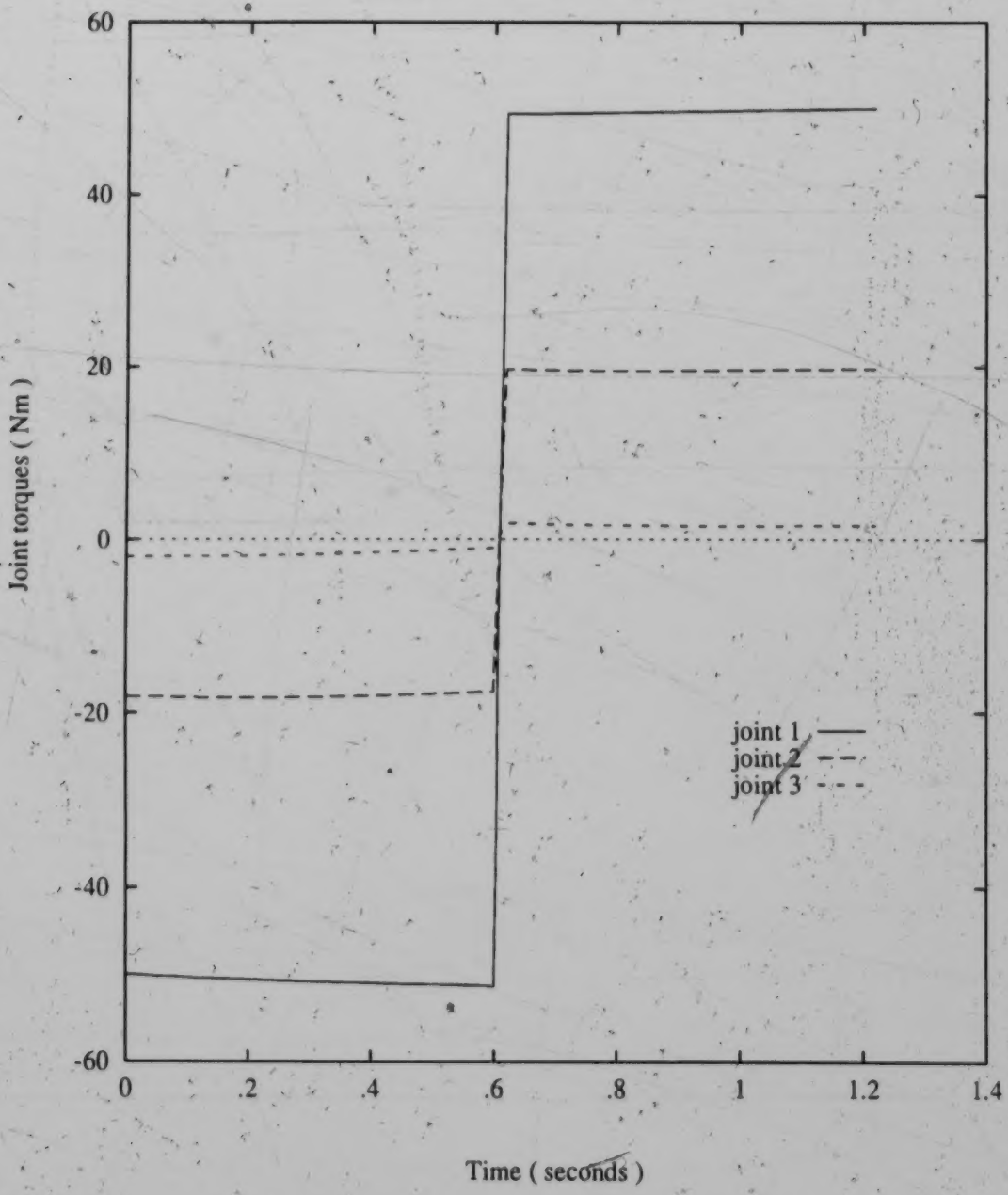
60
40
20
0
-20
-40
-60
0
.2
.4
.6
.8
1
1.2
1.4
Joint torques (Nm)
Time (seconds)
joint 1
joint 2
joint 3

Figure 4.6 (c)

The phase plane trajectory and the maximum velocity curve are shown in Figure 4.5. The intersection is found to be at 0.6 seconds when x=0.236.

With the switching time found, the time optimal solution can be obtained by first integrating $g(x, \dot{x})$ from t=0 to t=0.6 second and then switching to integrate $f(x, \dot{x})$ from t=0.6 onward till the final destination is reached. The angles and velocities of all the joints are shown in Fig 4.6(a) and Fig 4.6(b) respectively. It is seen that the joints velocities are all zero at the desired end point. Since this is not real time control, the required torque is calculated from the dynamic equation given in (3.10) with the joint angles, velocities and accelerations all known from equations (4.11.4), (4.6.3) and (4.7.3) respectively. The calculated joint torques shown in Figure 4.6(c) are the typical bang bang control type. The optimal time found in this particular example is t=1.22 seconds.

The results of a second example are shown in figures 4.7 to 4.9(c). The path length, initial and final joint and Cartesian positions, the switching time, and the optimal time are all summarized in Table 4.1 below. In this second example, the end effector moves from P_0 to P_f form left to right as seen from Figure 4.7. The phase plane trajectories of the corresponding motion is shown in Figure 4.8. The joint angle, velocity and torque profiles are shown in Fig 4.9 (a),(b) and (c) respectively.

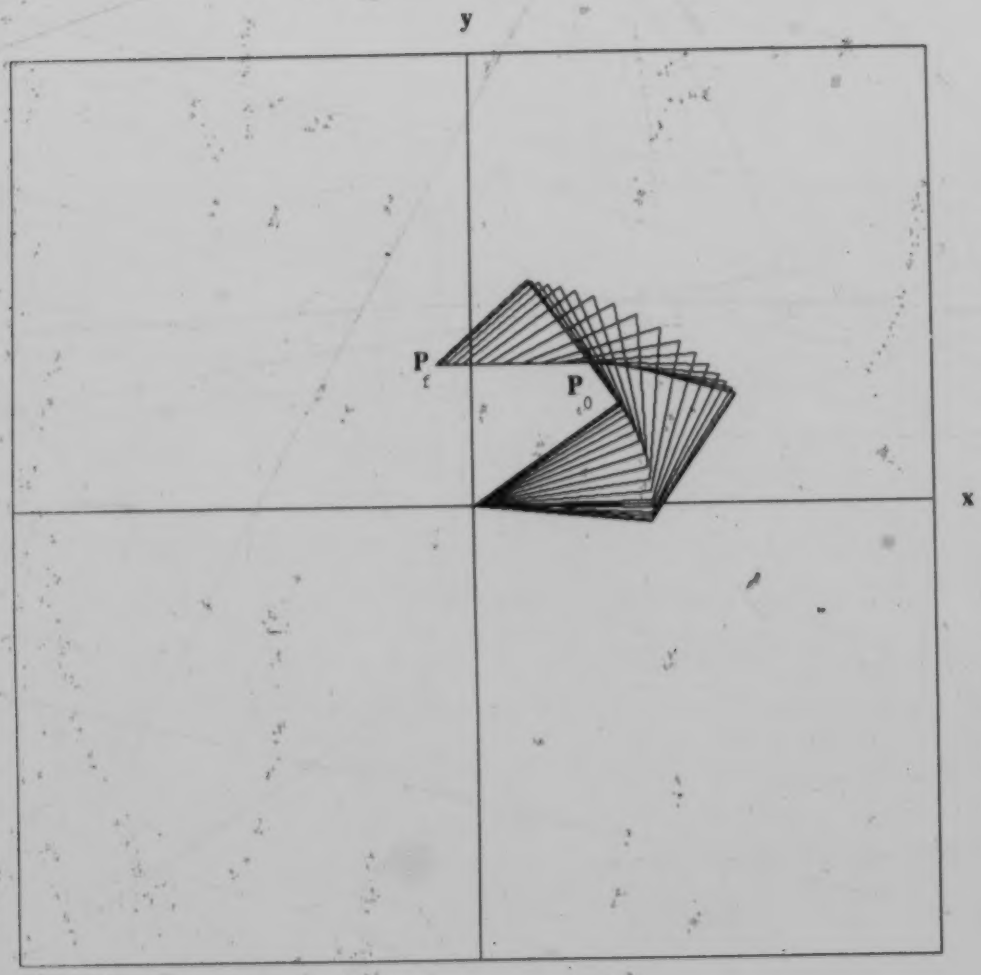

Figure 4.7 End effector motion for the second example

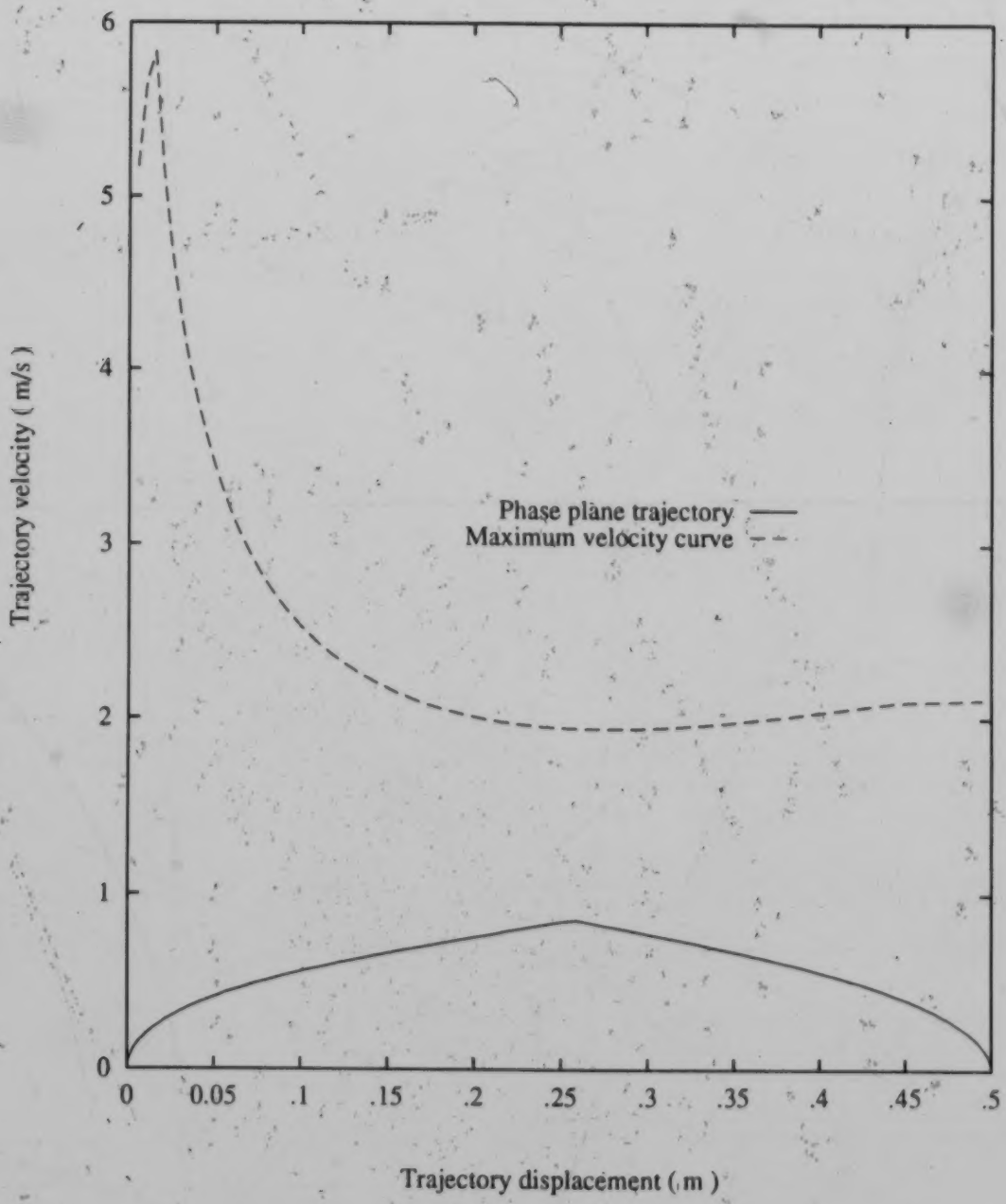

Figure 4.8 Phase plane trajectory for the second example

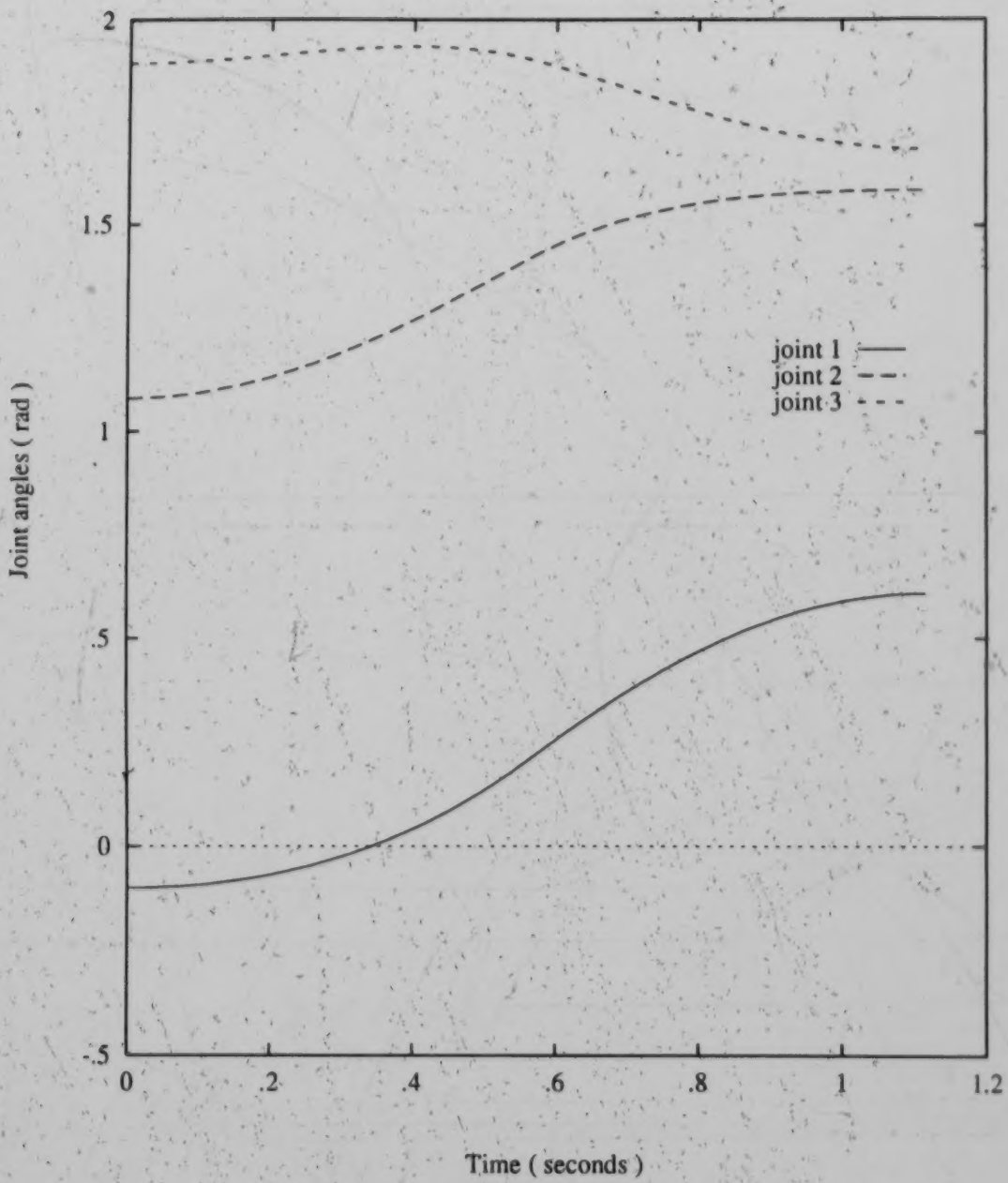

Figure 4.9 The resulting a) Joint angles, b) Joint velocities and c) Joint torques for the second example

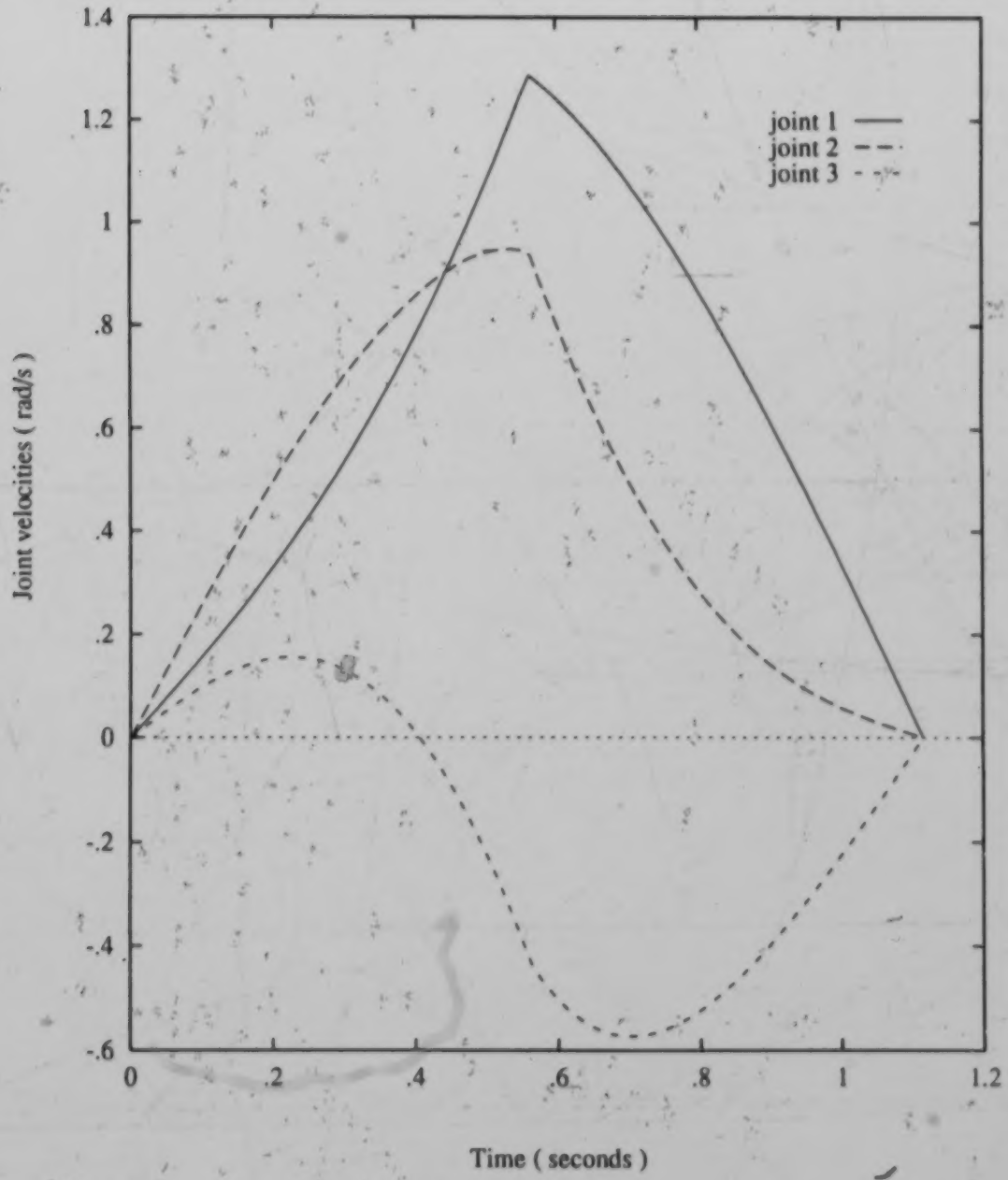
1.4
1.2
1
.8
.6
.4
.2
0
-.2
-.4
-.6
0
.2
.4
.6
.8
1
1.2
joint 1
joint 2
joint 3
Joint velocities (rad/s)
Time (seconds)

Figure 4.9 (b)

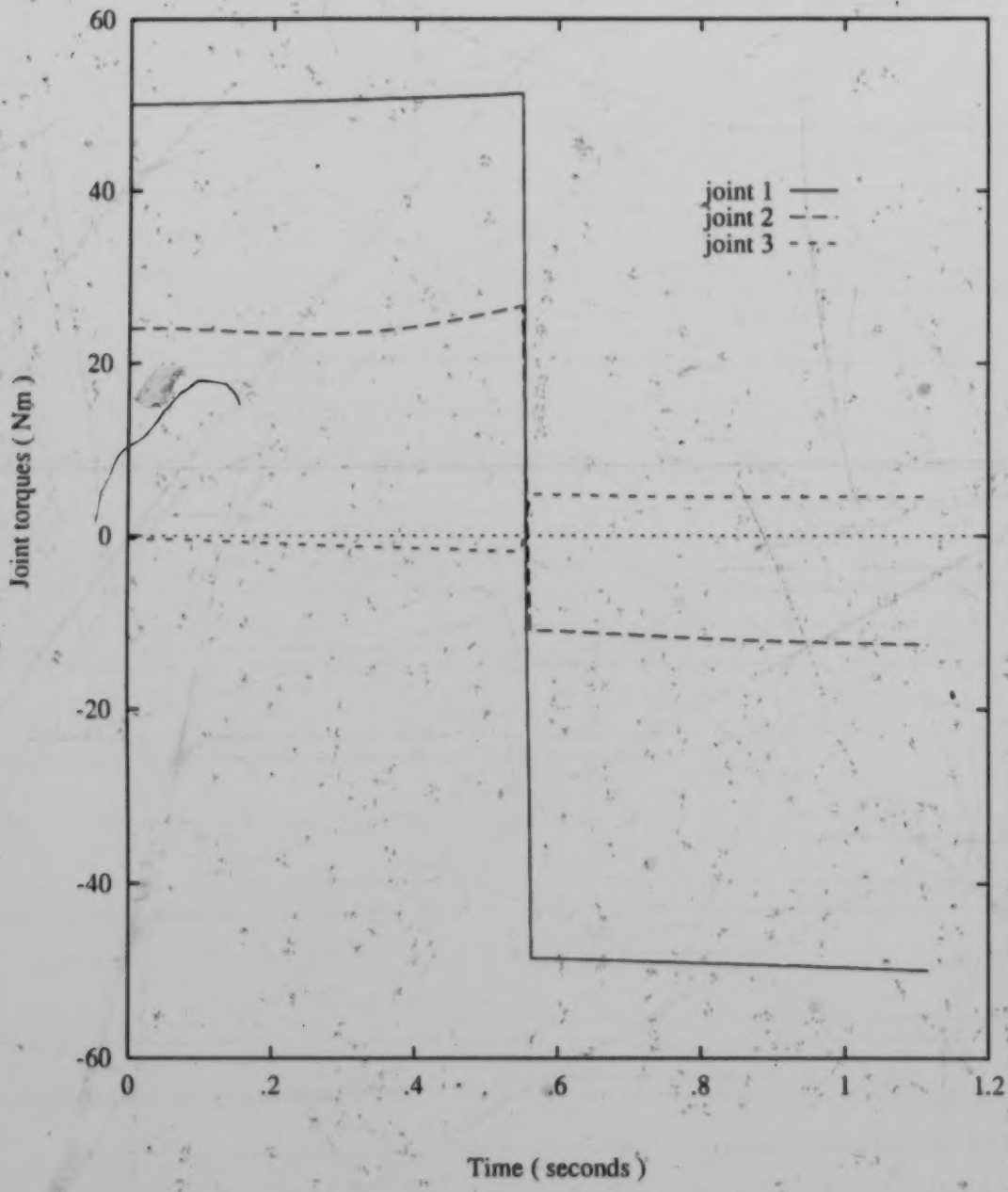
60
40
20
0
-20
-40
-60
0
.2
.4
.6
.8
1
1.2
joint 1
joint 2
joint 3
Joint torques (Nm)
Time (seconds)

Figure 4.9 (c)

Path length	0.5
Initial joint angles	(-0.1,1.079,1.89)
Final joint angles	(0.609,1.586,1.6885)
Initial Cartesian Position	(0.4,0.4)
Final Cartesian Position	(-0.1,0.4)
Switching time	0.56
Optimal time	1.16

Table 4.1 Numerical values of the second example

4.7 Multiple Switchings Case

The key to solving the time optimal problem is to find an admissible maximum or minimum acceleration that will yield a minimum time solution. It is important then to define a region where an admissible acceleration always exists such that the end effector remains on the specified path. Time optimal solution is possible only if condition $g(x, \dot{x}) \leq f(x, \dot{x})$ is satisfied throughout the optimal solution. This is best interpreted from the phase plane. The phase plane can actually be divided into two portions, one region where $g(x, \dot{x}) < f(x, \dot{x})$ is satisfied and the other region where $g(x, \dot{x}) > f(x, \dot{x})$. The maximum tolerable velocity curve is found by solving the equation $f(x, \dot{x}) = g(x, \dot{x})$ and a typical multiple switching curve is shown in Figure 4.10. It is seen from the maximum velocity curve that condition $f(x, \dot{x}) = g(x, \dot{x})$ is transformed to a bound on the velocity $\dot{x}$ in the phase plane. For, when the velocity $\dot{x}$ is above the maximum velocity curve, the end effector cannot remain on the path with any combination of admissible torques.

The multiple switchings method is used only when single switching fails to provide a valid time optimal solution. This occurs when the forward integration of

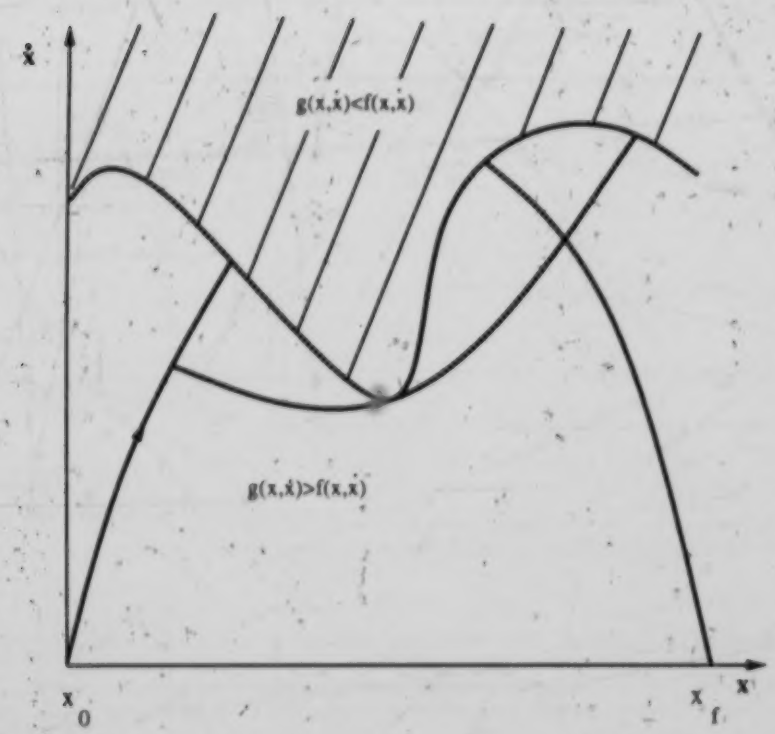

Figure 4.10 Multiple Switching Curve

maximum acceleration from x_0 and the backward integration of maximum deceleration from x_f do not intersect before their velocities becomes so large that $f(x,\dot{x}) \geq g(x,\dot{x})$. In that case, the phase plane trajectories of the maximum acceleration in the forward direction and maximum deceleration in the backward direction intersect the maximum velocity curve before they intersect each other. Instead of a single switch to the deceleration driving the end effector to the destination x_f, multiple switches between accelerations and decelerations have to be made earlier to prevent the trajectory velocity from becoming too large and entering the prohibited region in the phase plane.

Examples for the multiple switching are not given here for it has been very difficult to find examples which require multiple switchings. A couple of examples have actually been found when the manipulator is given a very awkward starting position. These examples, however, have a v shape maximum velocity curve and hence violate the assumption of the switching algorithm that the maximum velocity curve must be smooth at its minimum point [Shiller 90].

Chapter 5

An Obstacle Avoidance Algorithm

5.1 Introduction

In this chapter, the use of redundant manipulators for obstacle avoidance is discussed. As mentioned earlier, in the case of a non-redundant manipulator one can have algorithms that generate either a time optimal motion along a specific path or a minimum time obstacle avoidance trajectory but not both. It is shown here that the extra degree of freedom of a redundant manipulator can be fully utilized to produce a minimum time specified path and also to avoid obstacle at the same time.

An obstacle avoidance algorithm based upon the inverse kinematic function is developed here and incorporated into the time optimal algorithm presented in chapter 4. The inverse kinematic function approach was first used to add constraints to redundant manipulators [Wampler II 87]. An inverse kinematic function defined in terms of the location of the obstacle is used to find a set of joint trajectories that avoids an obstacle in the workspace. This avoidance algorithm is then incorporated into the time optimal algorithm to generate a minimum time obstacle avoidance algorithm.

5.2 Inverse Kinematic Function Approach

The simplest and easiest way to define a function for an underdetermined system is to add additional constraints to make the system well determined and to satisfy the

desired objectives at the same time. Defining the inverse kinematics as a function for a redundant system is essentially the same as adding extra equations to the system. The number of additional equations needed is the same as the degree of redundancy of the manipulator. Since the planar redundant manipulator discussed here has one extra joint, it is only required to define one additional function. Intuitively, this chosen function must contain some information about the location of obstacle. Detailed description of the additional function for the three link planar redundant manipulator is given below, followed by the kinematic solution and an example.

5.2.1 Formulation of the Additional Function

The only information about the obstacle required in the definition of the additional function is a point in the workspace which can represent the obstacle. This point is chosen according to the shape and the size of the obstacle. For an obstacle with irregular shape, this point can be chosen as a point which is nearest to the manipulator or a point on the obstacle which is likely to be obstructing the manipulator. However, for an obstacle which has a small size, it can be taken as the center of the obstacle. This point is denoted as (x_c, y_c) on the Cartesian plane in the subsequent derivation of the algorithm.

A function of the joint angles $F(\theta)$ can be defined as

$$F(\theta)=\sum_{i=1}^{n}\alpha_i s_i^{a_i}, \qquad n=3 \tag{5.1}$$

where α_i and a_i are constants to be specified. n is the total number of joints of the redundant manipulator and is equal to 3 in this case. s_i's are the distances between (x_c, y_c) and the end point of each link. It is written as

$$s_i = ((x_i - x_c)^2 + (y_i - y_c)^2)^{\frac{1}{2}} \tag{5.2}$$

where x_i and y_i are the Cartesian positions of the end point of link i. They can be obtained easily from the geometry of the manipulator as

$$(x_1 , y_1) = (l_1 c_1 , l_1 s_1) \tag{5.3.1}$$

$$(x_2 , y_2) = (l_1 c_1 + l_2 c_{12} , l_1 s_1 + l_2 s_{12}) \tag{5.3.2}$$

$$(x_3 , y_3) = (l_1 c_1 + l_2 c_{12} + l_3 c_{123} , l_1 s_1 + l_2 s_{12} + l_3 s_{123}) \tag{5.3.3}$$

The function $F(\theta)$ can be expressed in terms of the joint angles when equations (5.3) are substituted back into (5.1). Denoting V_i as the square of the distance s_i for the i^{th} joint, $F(\theta)$ can be written as

$$F(\theta) = \alpha_1 V_1^{\frac{a_1}{2}} + \alpha_2 V_2^{\frac{a_2}{2}} + \alpha_3 V_3^{\frac{a_3}{2}} \tag{5.4.1}$$

where

$$V_1 = l_1^2 + x_c^2 + y_c^2 + P_1 \tag{5.4.2}$$

$$V_2 = V_1 + l_2^2 + 2 l_1 l_2 c_2 + P_2 \tag{5.4.3}$$

$$V_3 = V_2 + l_3^2 + 2 l_1 l_3 c_{23} + 2 l_2 l_3 c_3 + P_3 \tag{5.4.4}$$

The P_i are again substitutions used to keep the size of the above expressions reasonable and the remaining derivation simpler. They are defined as

$$P_1 = -l_1(c_1 x_c + s_1 y_c) \tag{5.4.5}$$

$$P_2 = -l_2(c_{12} x_c + s_{12} y_c) \tag{5.4.6}$$

$$P_3 = -l_3(c_{123} x_c + s_{123} y_c) \tag{5.4.7}$$

With the above equations, the additional function is completely defined. α_i and a_i in (5.4.1) are constants to be chosen for the control of individual joint. The function $F(\theta)$ in (5.1) can be chosen as the sum of the distances between the obstacle and the end point of each link or it can be the sum of the reciprocals of the distances. It depends upon the sign of the constant a_i. α_i can even be chosen as any number other than unity, thus $s_i^{a_i}$ is not limited to be the distance but rather the distance raised to an arbitrary power. α_i can be interpreted as the proportional constant or weighting factor for its corresponding distance s_i from the obstacle. These two constants provide certain degree of freedom for the variation of the function $F(\theta)$. It allows different weights to be put on particular joint to achieve best obstacle avoidance joint geometry.

A third equation in addition to the kinematic equations (3.1) is actually formed by equating $F(\theta)$ to a polynomial of time. The new equation can be written as

$$F(\theta) = \sum_{i=0}^{m} \beta_i t^i \tag{5.5}$$

where m is the order of the polynomial and β_i's are its coefficients. $F(\theta)$ is being equated to a polynomial in t for it is desirable to vary the distances between the obstacle and the manipulator as a general function of time. The joints can be made to stay away from the obstacle with proper choice of the coefficients β_i and the order m of the polynomial. For $m=0$, the sum of the distances between the joints and the obstacle

has to be unchanged throughout the whole trajectory and equal to β_0. This is obviously not the best choice for the end effector can only move for a very short distance before this requirement is violated. When m=1, the distances are required to vary as a linear function of time. This is, of course, a better choice over the previous one for the end effector can now travel for a longer distance before its position is violated eventually. A second order polynomial is used for the discussion below because it replaces the straight line velocity-obstacle distance profile by a parabolic one. It can level off the increase of the distances with appropriate choice of β_2. Higher order polynomials are not considered simply because it is not necessary and the choice of β_i's are substantially more complicated.

5.2.2 Inverse Kinematic Solutions

The basic inverse kinematic algorithm developed in chapter four for redundant manipulator is used here to obtain the inverse kinematic solution. The dynamics of the manipulator is not taken into consideration at this stage. With the inclusion of the additional equation, the solution of the inverse kinematic problem becomes straight forward since the use of the pseudoinverse is no longer necessary. However, closed form solution is still far from being realizable because of the complexity of the additional equation and its dependence upon t. The possibility of solving the kinematics numerically as a non-linear system is ruled out because of the difficulties involved in numerical iterations and the convergence to the desired solution. The inverse kinematic algorithm developed in chapter four, however, can be modified to include the extra equation for obstacle avoidance. The advantage is that no numerical iterations are required and continuity can be detected by having a maximum allowable change in all the joint angles.

It is seen from chapter four that the pseudoinverse is used in equation (4.11.2) to find the incremental change of the joint angles. To find the Jacobian of the kinematic equations, the gradient of the function $F(\theta)$ has to be evaluated. Again, further substitutions are used to keep the expressions short and facilitate the computer implementation. Equation (5.4.1) is used to take the partial derivatives which are denoted by a second subscript appended to the V_i to form V_{ij}'s terms where j is the joint angle that the partial derivative is taken with respect to. The partial derivatives of the P_i's terms are represented likewise. Using the above defined notation, the partial derivatives of $F(\theta)$ with respect to all the joint angles are given below

$$\frac{\partial F(\theta)}{\partial \theta_1} = \sum_{i=1}^{3} K_i V_{i1} \tag{5.6.1}$$

$$\frac{\partial F(\theta)}{\partial \theta_2} = \sum_{i=2}^{3} K_i V_{i2} \tag{5.6.2}$$

$$\frac{\partial F(\theta)}{\partial \theta_3} = K_3 V_{33} \tag{5.6.3}$$

where

$$K_i = \frac{\alpha_i a_i}{2} V_i^{\left(\frac{a_i}{2}-1\right)} \qquad i=1,2,3 \tag{5.6.4}$$

The partial derivatives of V_i are given in terms of one another and in terms of the partial derivatives of P_i's. They are written as

$$V_{11} = P_{11} \tag{5.7.1}$$

$$V_{21} = V_{11} + P_{21} \tag{5.7.2}$$

$$V_{31} = V_{21} + P_{31} \tag{5.7.3}$$

$$V_{22} = -2l_1l_2s_2 + P_{22} \tag{5.7.4}$$

$$V_{32} = V_{22} - 2l_1l_3s_{23} + P_{32} \tag{5.7.5}$$

$$V_{33} = -2l_1l_3s_{23} - 2l_2l_3s_3 + P_{33} \tag{5.7.6}$$

From (5.4.5) only the partial derivative of P_1 with respect to joint one is non-zero and for P_2, the partial derivatives with respect to joint one and two are identical. All the partial derivatives for P_3 are the same. Therefore, there are only three distinct P_{ij} terms and they are given as

$$P_{11} = 2l_1(s_1x_c - c_1y_c) \tag{5.7.7}$$

$$P_{21} = 2l_2(s_{12}x_c - c_{12}y_c) \tag{5.7.8}$$

$$P_{31} = 2l_3(s_{123}x_c - c_{123}y_c) \tag{5.7.9}$$

The first two rows of the Jacobian matrix remain the same as in (3.1.2) and the third row is the gradient of $F(\theta)$ given in equations (5.6) with the substitutions of (5.7). The same substitutions as in (5.7) are used for computer implementation of the Jacobian. Since the dimension of the Jacobian inverse in equation (4.11.2) is changed from 3×2 to 3×3, the corresponding dimension of $d\mathbf{r}$ has to be changed also. The first two entries of (4.11.3) remain the same; however the last entry is the time derivative of the right hand side of equation (5.5). From the original definition of $d\mathbf{r}$ given in (4.6.1), the unit vector $\mathbf{v}$ is formed by factoring out the magnitude of the trajectory

velocity $\dot{x}$. If the first two entries of the new v vector are to be unchanged, the last element of the v vector becomes the last entry of $d\mathbf{r}$ vector divided by the trajectory velocity $\dot{x}$. Doing this creates problems at the starting and finishing points of the trajectory when the velocities are zeros. This problem is avoided by setting this last entry to zero whenever $\dot{x}$ is zero. This is justified because $\dot{x}$ is to be multiplied back to v as in (4.7.3). Subroutines in chapter four are modified in accordance with the above changes to obtain the inverse kinematic solution for the obstacles avoidance case.

5.2.2.1 Choice of Obstacle Avoidance Constants

Even with the simple function $F(\theta)$, there is already a large variety of possible combinations of the choices of α_i and a_i. Obstacle avoidance can only be achieved with proper choices of these constants and it is impossible to find them by trial and error. It is attempted here to provide some guidelines for the determination of the constants to generate a desirable obstacle avoidance trajectory. The overall effects of the six constants α_i and a_i on the joints are rather difficult to comprehend at one time. It is easier to consider the effects of the corresponding constants on a single joint first and then include the other joints for the overall effects.

The effects of the constants for joint two and three are eliminated by setting both α_2 and α_3 to zeros. As a result, $F(\theta)$ becomes a function of the distance between the end point of joint one and the obstacle only. The addition equation becomes

$$\alpha_i s_1^{a_1} = \beta_0 + \beta_1 t + \beta_2 t^2 \qquad (5.8.1)$$

To study the effect of the constants in (5.8.1) on s_1, equation (5.8.1) is differentiated to investigate the incremental change of s_i at each time increment.

Differentiate (5.8.1) with respect to time

$$\frac{d\,\alpha_1 s^{a_1}}{dt} = \beta_1 + 2\beta_2 t \qquad (5.8.2)$$

$$\Rightarrow \qquad \alpha_1 a_1 s_1^{a_1-1} ds_1 = (\beta_1 + 2\beta_2 t)dt \qquad (5.8.3)$$

The change in time dt and the change in distance ds_1 are considered as small incrementals δt and δs_1 respectively. Equation (5.8.3) can be rewritten as follow with δs_1 as the subject

$$\delta s_1 = \frac{(\beta_1 + 2\beta_2 t)\delta t}{\alpha_1 a_1 s_1^{a_1-1}} \qquad (5.8.4)$$

The incremental change of the obstacle distance from joint one is made subject of the formula because it represents the effectiveness of the algorithm. The larger is δs_1 for each time step increment, the further away the joint is moving away from the obstacle. The choice of the constants can now be made according to (5.8.4). The criteria for choosing the constants in the denominator of (5.8.4) is first discussed for it can only affect joint one. The choice of the β's will be discussed later because it has a global effects on all joints.

It is noted that α_1 affects δs_1 independent of the current obstacle distance δs_1 whereas the effect of a_1 on δs_1 is dependent on s_1 and the discussion is much more involved. The effect of α_1 is perhaps the most obvious, it is inversely proportional to δs_1. However, when the end point of a particular joint is moving toward the obstacle, it should be set to negative. A negative δs simply means that the end point of the

joint is moving away from the obstacle instead of moving towards it. The effect of varying a_1 on δs_1 is rather difficult to know from (5.8.4). Therefore, the terms that contain a_1 are isolated and the following function is formed

$$h(a_1,s_1) = \frac{1}{a_1 s_1^{a_1-1}} \tag{5.8.5}$$

where $h(a_1,s_1)$ is taken as a two independent variable function of a_1 and s_1. The dependence of a_1 on s_1 implies that the effect of a_1 on δs_1 varies along the trajectory as the end point of joint one moves away or towards the obstacle. Equation (5.8.5) is plotted as a single independent variable function at different values of s_1. The series of curves are shown in Figure 5.1. Similarly, another series of curves are also plotted at different values of a_1 as in Figure 5.2. It is seen from Figure 5.2 that a_1=1 is a special case where s_1 has no effect on δs_1 at all. This is useful for the selection of the constants β_1 and β_2. It is desirable to know the approximate range of the obstacle distance s_1 for which the value of $h(a_1,s_1)$ varies substantially. With the priori knowledge of s_1, a_1 is chosen in such a way that the desirable values of h lies within the approximate boundaries of the obstacle distance curves shown in Figure 5.1. The major advantage of using (5.8.5) for obstacle avoidance is seen in the following.

From Figure 5.1, it is found that the variation of h within a certain range of s_1 for $0<a_1<1$ and $1<a_1<2$ are very close. However, from Figure 5.2, the shape of the curves for $a_1>1$ is drastically different from that for $a_1<1$. For $a_1>1$, as joint one is moving away from the obstacle h diminishes rapidly, that is the increment δs_1 decreases. This is highly desirable because if the end point of joint one keep moving away from the obstacle at a constant rate, the end effector position will no longer be satisfied after a short while. On the contrary, when $a_1<1$, h increases as the end point

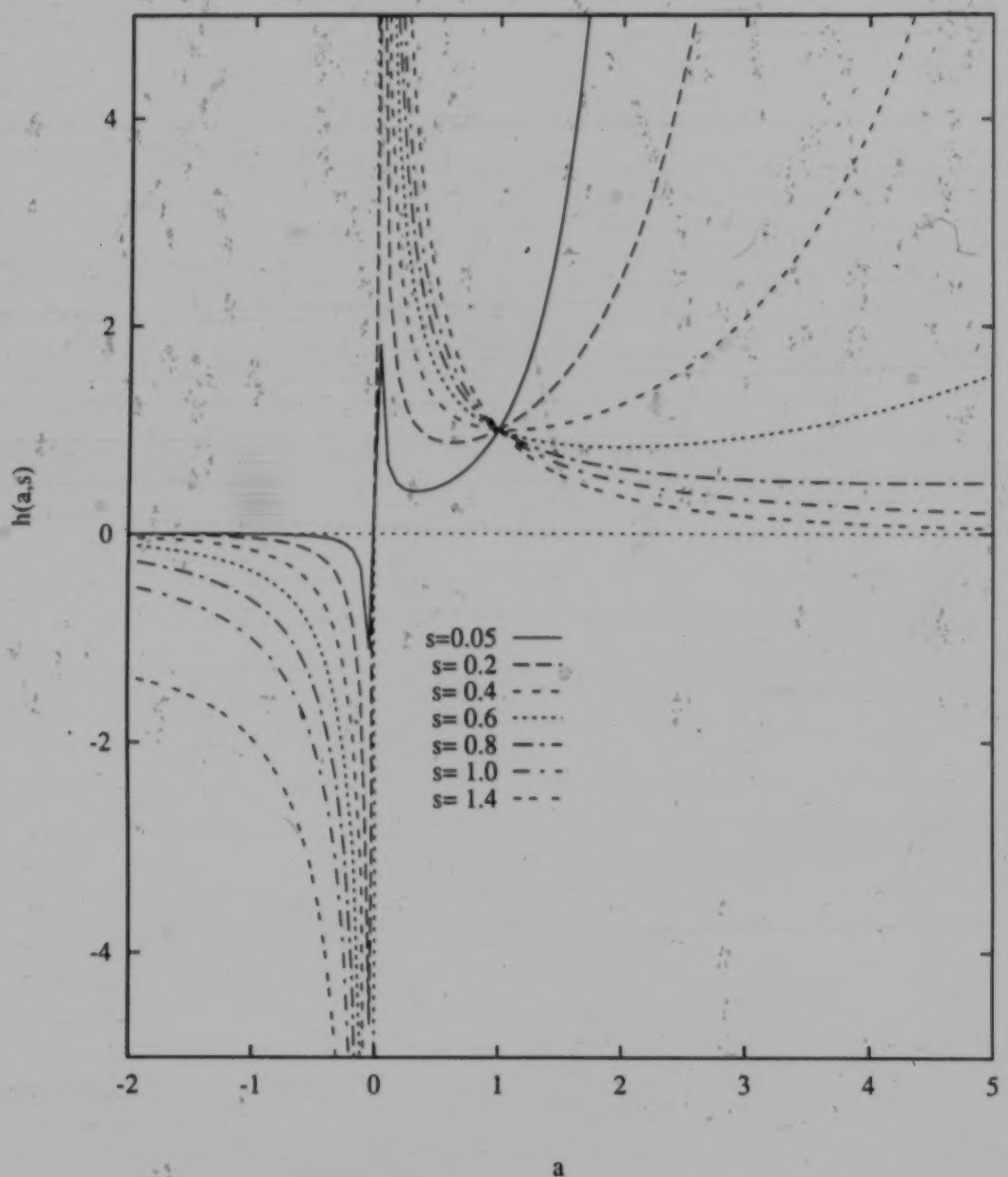

Figure 5.1 Effect of s on h(a,s)

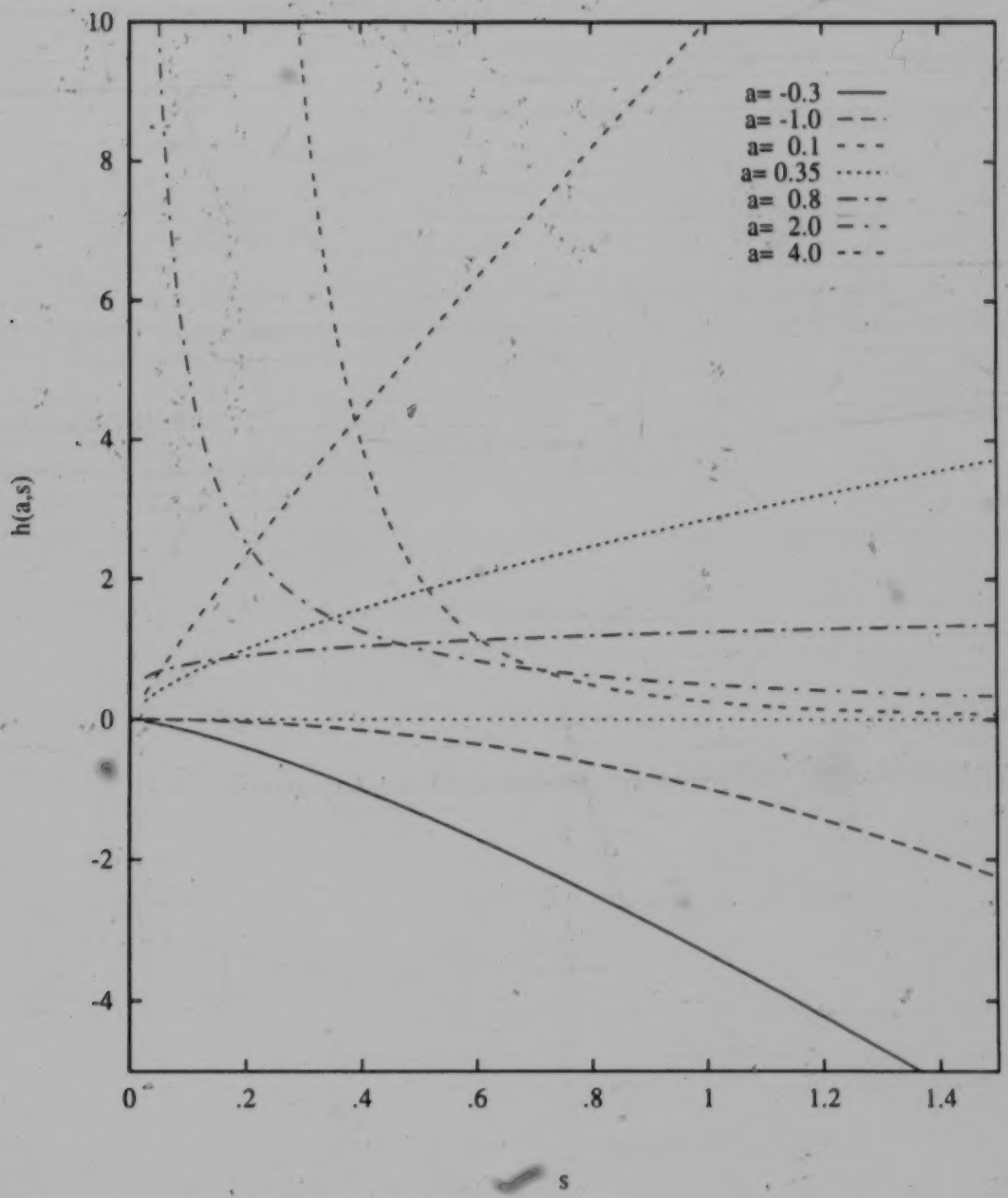

Figure 5.2 Effect of a on h(a,s)

moves away. This is equally desirable because there are situations where the end effector has to move towards the obstacle temporarily in the course of avoiding the obstacle or to follow the specified path when the joint is the end effector. In that case, a_1 is best chosen to be less than unity. It is noted from Figure 5.1 that there is a singular point at a_1=0 which should be avoided. Also, when a_1 is negative, the behavior of h is similar to that for $0<a_1<1$ except that it is negative. If h is desired to be negative, the proportional constant α_1 can be set to negative. Therefore, it is concluded that there is no need to use a negative a_i. Figure 5.2 is mainly used to see the actual range of h after a_1 has been chosen from the above guidelines.

The remaining constants to be determined in equation (5.8.4) are the β's. These values depend strongly upon the location of the obstacle and the posture of the manipulator. There are situations in which they are not required and are set to zeros. For when they are needed, there is not much room for the choice for their magnitude. It is known that β_1 is actually the velocity of the sum of the distance terms moving away from the obstacle. If β_1 is too large, the end effector is incapable of following the designated path, and if it is too small, the effect of it cannot be seen. If the obstacle is successfully avoided after setting a proper value of β_1, β_2 is set to zero. When the desired path cannot be satisfied with any choice of β_1, β_2 is used to level off the constant increase of δs_1 which drive the end effector off track. If that is the case, a negative β_2 is used and its magnitude is adjusted to get the best result.

Because the joint angles are strongly coupled with each other, the control of joints two and three are in general more difficult than control of joint one. The general approach to find all the constants is that

- Set β_1 and β_2 as zeros and the rest of the constants to one's except for the proportional constants α_3. If the specified path of the end effector is moving toward the obstacle, α_3 is set to be negative one otherwise α_3 is set to one. Generate the

inverse kinematic solution and compare the behavior of the intermediate joint motions with the kinematic solution for no obstacle case.

- Set β_1 and β_2 according to the result from the previous step. β_1 is chosen as a small positive number first to have the joints move away from the obstacle at a constant speed. Then, if the end effector deviates at the end of the trajectory, β_2 is increased until the best result is obtained.
- This procedure is only required if the above two steps cannot provide an acceptable solution. The constants α_i and a_i are adjusted together in pair to modify the behavior of all the joints. The desired motions of the joints are achieved by varying the constants according to above given properties of the function h.

The above rules serve only for general guidance and they may have to repeated in order to refine the choice of the constants. There is no single unique combination of the constants to get the best result and the same result may be obtained with different combination of the constant. There are other factors that can directly affect the performance of this algorithm.

Two major factors that influence the obstacle avoidance solution are the location of the obstacle and the starting joint position of the manipulator. However, there are some general restrictions that can be used to ensure proper behavior of the inverse solution. The obstacle center cannot be placed anywhere on or in the vicinity of the trajectory if both obstacle avoidance and the path requirement are to be satisfied. Putting the obstacle too close to the end point of any intermediate joint is also to be avoided for it causes singularity. A certain minimum distance from the obstacle is preferable. For cases where the obstacle is too near to a particular joint, an alternate position which is further away than the real obstacle center may be used to avoid violating the above restriction and yield a better solution.

5.2.3 Example

The first example given in chapter four is used again here to demonstrate the obstacle avoidance algorithm. The center of the obstacle is placed at (0.6, –0.3) which is chosen deliberately to be quite close to the trajectory and it cannot be placed much closer without making the choice of the constants very difficult. As seen from position of the obstacle, the end point of joint three or the end effector is moving toward the obstacle. a_3 is therefore set to negative with both β_1 and β_2 set to zeros from the above guidelines. The result is shown in Figure 5.3 and it is observed that joint one has changed substantially and the end effector position requirement is not met near the end of the trajectory. Moreover, the obstacle is still not avoided, therefore a small value of β_1=0.0015 is used and the obstacle is successfully avoided as seen in Figure 5.4. In this example, it is not necessary to use non-zero β_2 and non-unity values of α_i and a_i to enhance or suppress individual joint motion.

5.3 Incorporation of the Obstacle Avoidance Algorithm

The above derived inverse kinematic solution can be incorporated to the minimum time algorithm only with some modification. Since the Jacobian is no longer rectangular and the foregoing formulation of the obstacle avoidance case resembles a lot the non-redundant case. The joint velocities are obtained from equation (4.6.3) and the required Jacobian has already been derived in the inverse kinematic solution in chapter 3. The joint acceleration are obtained from (4.7.3). In (4.7.3), the unknown quantity yet to be derived is the derivative of the Jacobian. The first two rows of $\dot{J}(\theta)$ remains unchanged and the elements in the last row can be found according to the definition (4.7.2). Renaming the partial derivative of $F(\theta)$ with respect to joint i in equations (5.6) as u_i, the last row of the derivative of the Jacobian is written as

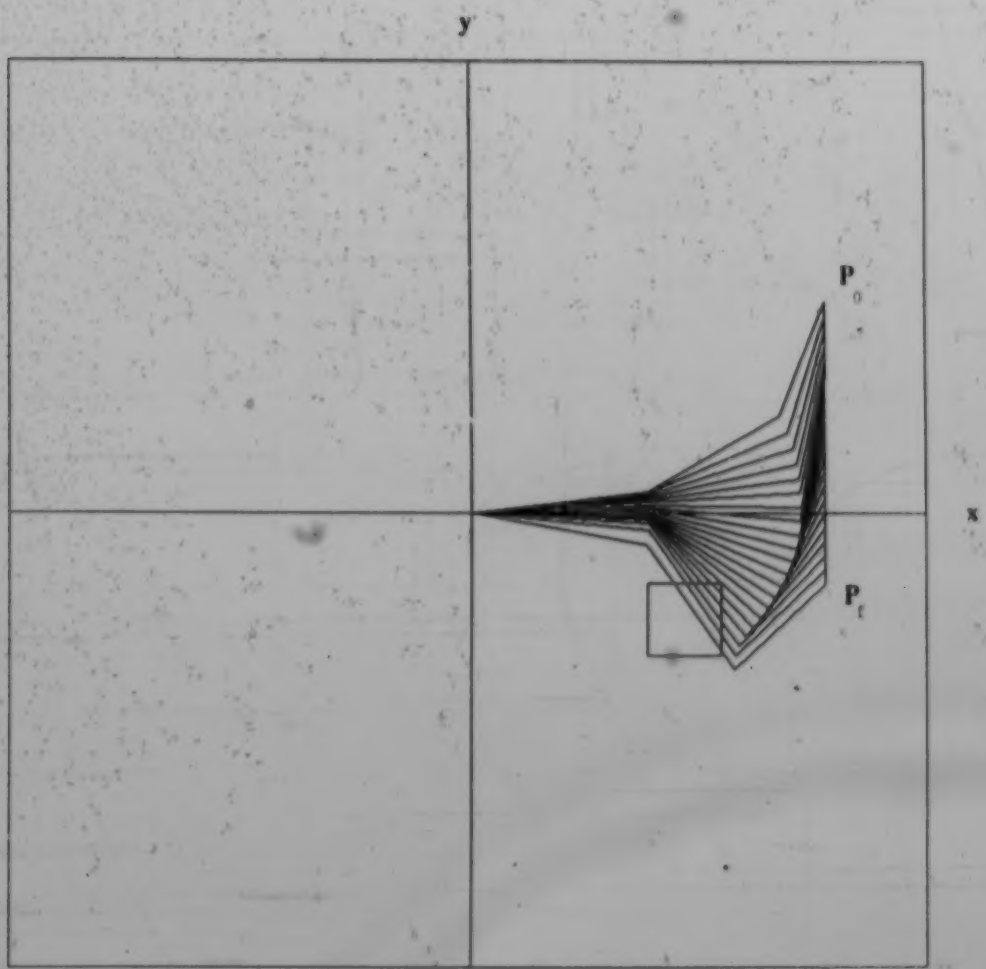

Figure 5.3 Intermediate Result of Obstacle Avoidance Example

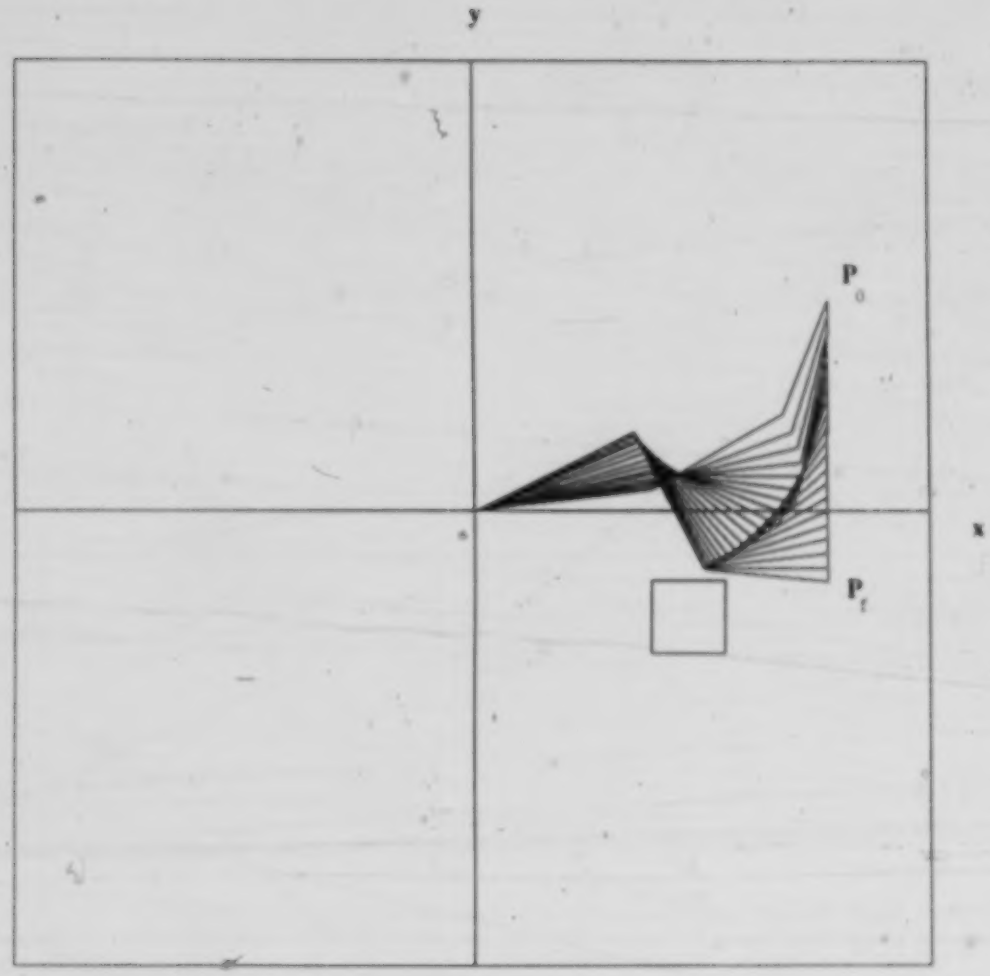

Figure 5.4 Final Result of Obstacle Avoidance Example

$$\left[\sum_{i=1}^{3} \frac{\partial u_1}{\partial \theta_i}\dot{\theta}_i \qquad \sum_{i=1}^{3} \frac{\partial u_2}{\partial \theta_i}\dot{\theta}_i \qquad \sum_{i=1}^{3} \frac{\partial u_3}{\partial \theta_i}\dot{\theta}_i \right] \tag{5.9.1}$$

The same substitutions in the Jacobian of $F(\theta)$ are also used here and an additional subscript is again used for the partial derivatives of V_{ij}'s in (5.7). Taking the partial derivatives in (5.9.1) and after simplification its transpose can be written as follow

$$\begin{bmatrix} \sum_{i=1}^{3}(q_i V_{i1}^2 + K_i V_{i11})\dot{\theta}_1 + \sum_{i=2}^{3}(q_i V_{i1} V_{i2} + K_i V_{i12})\dot{\theta}_2 + (q_3 V_{31} V_{33} + K_3 V_{313})\dot{\theta}_3 \\ \sum_{i=2}^{3}[(q_i V_{i1} V_{i2} + K_i V_{i12})\dot{\theta}_1 + (q_i V_{i2}^2 V_{i1} + K_i V_{i22})\dot{\theta}_2] + (a_3 V_{32} V_{13} + K_3 V_{323})\dot{\theta}_3 \\ (q_3 V_{31} V_{33} + K_3 V_{313})\dot{\theta}_1 + (q_3 V_{32} V_{33} + K_3 V_{323})\dot{\theta}_2 + (q_3 V_{33}^2 + K_3 V_{333})\dot{\theta}_3 \end{bmatrix} \tag{5.9.2}$$

where

$$q_i = \left(\frac{a_i}{2} - 1\right)\frac{K_i}{V_i} \qquad i = 1,2,3 \tag{5.9.3}$$

The derivatives of the V_{ij} terms are written in terms of the P_i terms defined in equations (5.4). Again, the later V_{ijk} terms are defined using the previously defined V_{ijk} terms to shorten the expressions. The derivatives of the V_{ij} terms are given as

$$V_{111} = -P_1 \tag{5.9.4}$$

$$V_{211} = V_{111} - P_2 \tag{5.9.5}$$

$$V_{311} = V_{211} - P_3 \tag{5.9.6}$$

$$V_{212} = -P_2 \tag{5.9.7}$$

$$V_{312} = V_{212} - P_3 \tag{5.9.8}$$

$$V_{313} = -P_3 \tag{5.9.10}$$

$$V_{222} = -2l_1l_2c_2 - P_2 \tag{5.9.11}$$

$$V_{322} = V_{222} - 2l_1l_3c_{23} - P_3 \tag{5.9.12}$$

$$V_{323} = -2l_1l_3c_{23} - P_3 \tag{5.9.13}$$

$$V_{333} = -2l_1l_3c_{23} - 2l_2l_3c_3 - P_3 \tag{5.9.14}$$

The derivatives of the Jacobian of $F(\theta)$ given above fully define (5.9.1) and is used as the last entry of the $\dot{J}$ matrix for the obstacle avoidance case. Because of the change in the dimension of the Jacobian, the dimension of both the tangential velocity v and acceleration vector w in (4.7.3) also has to be changed. Similar to the Jacobian, the first two entries remain unchanged as in (4.7.3). The joint velocity of the last term of $\dot{\mathbf{v}}$, however, is found by equating the right side of equation (4.6.2) to the derivative of the right side of (5.5). Denote the unit vector v by its components with subscripts x and y, equation (4.6.2) can be rewritten as

$$J(\theta)\dot{\theta} = \begin{bmatrix} v_x \\ v_y \\ \dfrac{\beta_1 + 2\beta_2 t}{\dot{x}} \end{bmatrix} \dot{x} \tag{5.10.1}$$

To find the joint accelerations, the derivative of the above equation has to be taken. The left hand side is unchanged and the right hand side is written as

$$\frac{d}{dt}\begin{bmatrix} v_x \\ v_y \\ \dfrac{\beta_1+2\beta_2 t}{\dot{x}} \end{bmatrix} = \begin{bmatrix} w_x\dot{x} \\ w_y\dot{x} \\ \dfrac{-(\beta_1+2\beta_2 t)\ddot{x}}{\dot{x}^2} + \dfrac{2\beta_2}{\dot{x}} \end{bmatrix}\dot{x} + \begin{bmatrix} v_x \\ v_y \\ \dfrac{\beta_1+2\beta_2 t}{\dot{x}} \end{bmatrix}\ddot{x} \qquad (5.10.2)$$

where w_x and w_y are the x and y components of the normal acceleration vector $\mathbf{w}$. It is seen from the above expression that the control variable $\ddot{x}$ appears also in the first term and this is due to the fact that $\dot{x}$ is in the denominator in (5.10.1). The separation of the control variable $\ddot{x}$ cannot be done directly. Since both $\dot{x}$ and $\ddot{x}$ are scalars, the terms in the last row can be rearranged. After simplification, equation (5.10.2) can be rewritten as

$$\begin{bmatrix} w_x \\ w_y \\ \dfrac{2\beta_2}{\dot{x}^2} \end{bmatrix}\dot{x}^2 + \begin{bmatrix} v_x \\ v_y \\ 0 \end{bmatrix}\ddot{x} \qquad (5.10.3)$$

The vectors $\mathbf{v}$ and $\mathbf{w}$ are modified as mentioned above to adjust to the size change of the Jacobian matrix. Knowing both $\mathbf{v}$ and $\mathbf{w}$, (4.7.3) can be used to found $\ddot{\theta}$. The derivation from (4.7.3) onward follows closely the non-redundant case presented in chapter four.

5.3.1 Example

The time optimal solution for the example given in section 5.2.3 is presented here. The inverse kinematic solution requires only single switching as can be seen from the maximum velocity curve and the forward and and backward integration curves in Figure 5.5. The intersection time is found to be $t = 0.467$ seconds at

$x = 0.194$ meters and the total time of travel is $t = 1.166$ seconds. The joint angles, velocities and torques profiles are shown in figures 5.6 (a), (b) and (c) respectively. It is noted that the optimal time obtained in this section is less than that in the no obstacle case. In the optimal algorithm developed in chapter 4, the inverse kinematic solution was obtained using the pseudoinverse, which is only one of the many possible solutions. As mentioned earlier, the optimal time is dependent upon the method used for solving the inverse kinematics; therefore the two results cannot be compared. It just happens that in this particular case the minimum time required for the obstacle free case is more than that in the case with the obstacle.

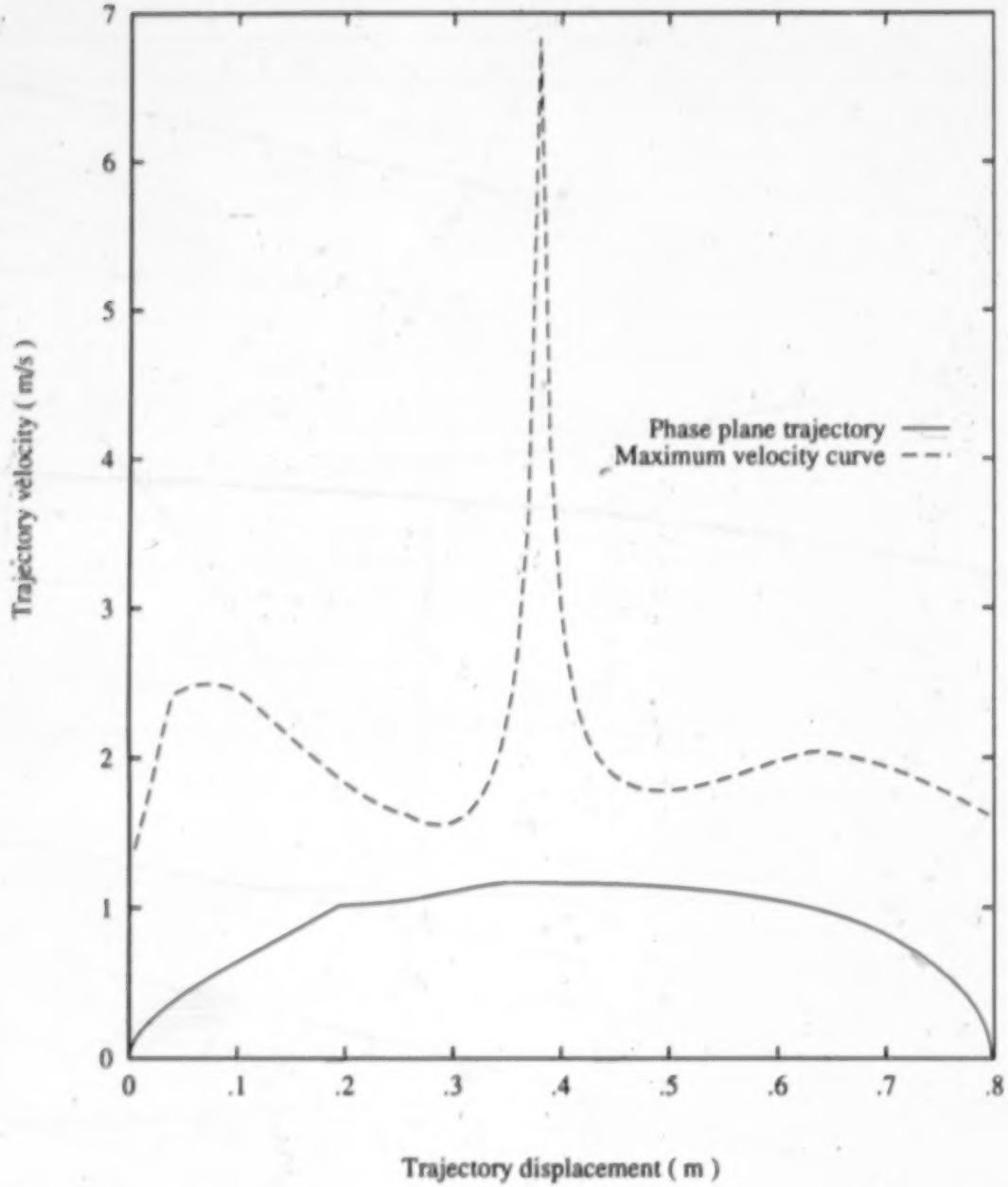

Figure 5.5 Phase plane trajectory for the time optimal obstacle example

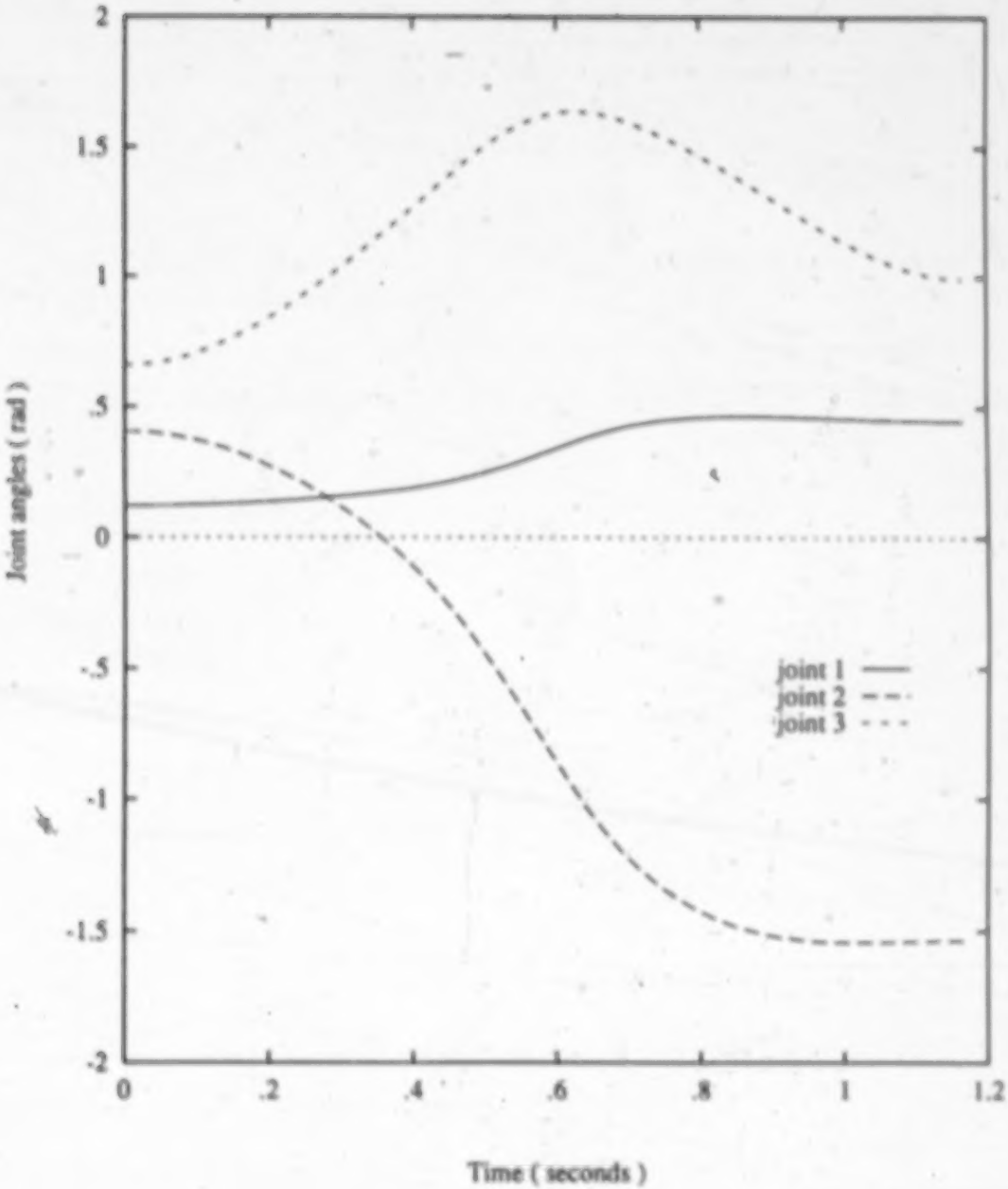

Figure 5.6 The resulting a) Joint angles, b) Joint velocities and c) Joint torques for the time optimal obstacle example

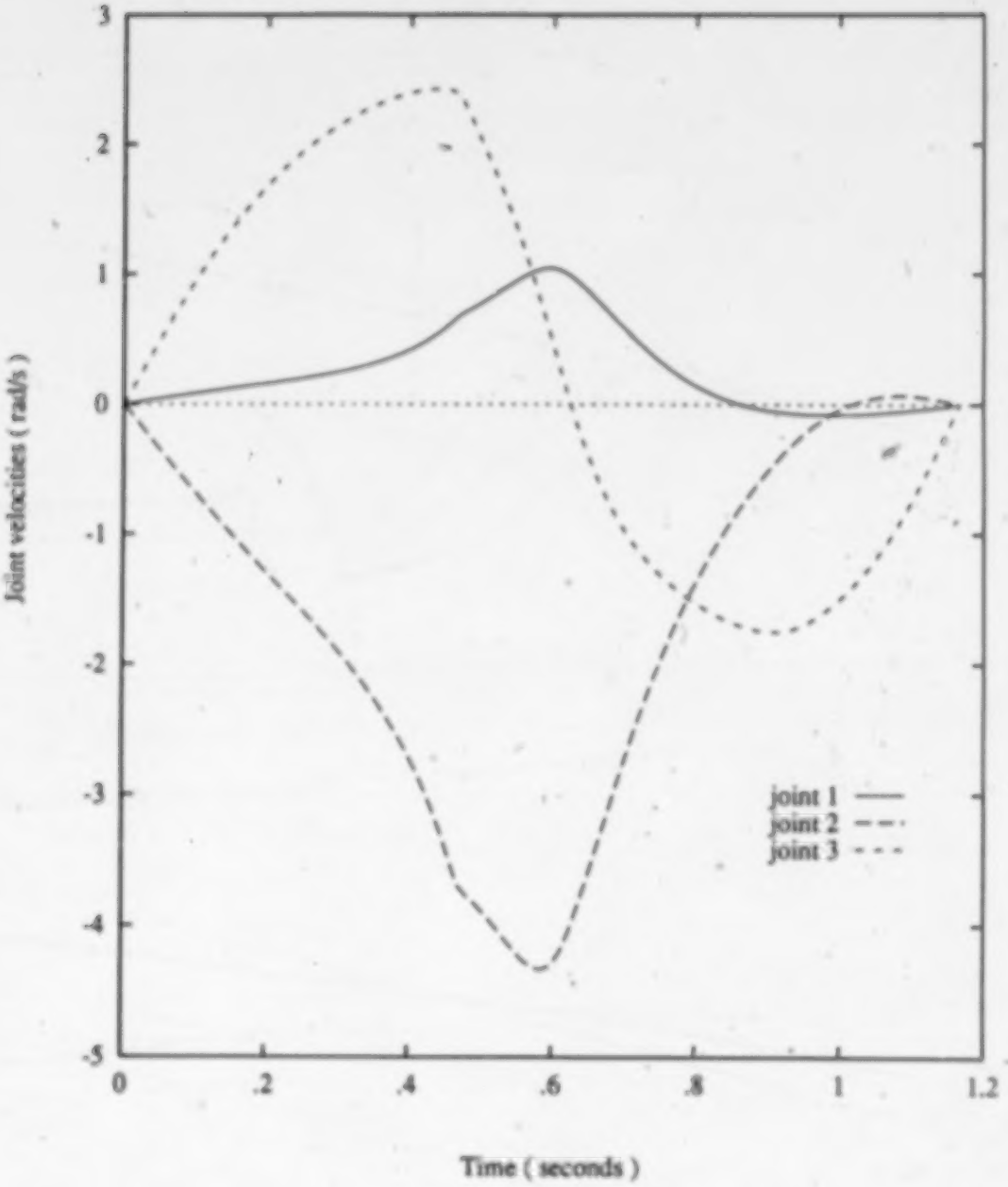
3
2
1
0
-1
-2
-3
-4
-5
0
.2
.4
.6
.8
1
1.2
Joint velocities (rad/s)
Time (seconds)
joint 1
joint 2
joint 3

Figure 5.6 (b)

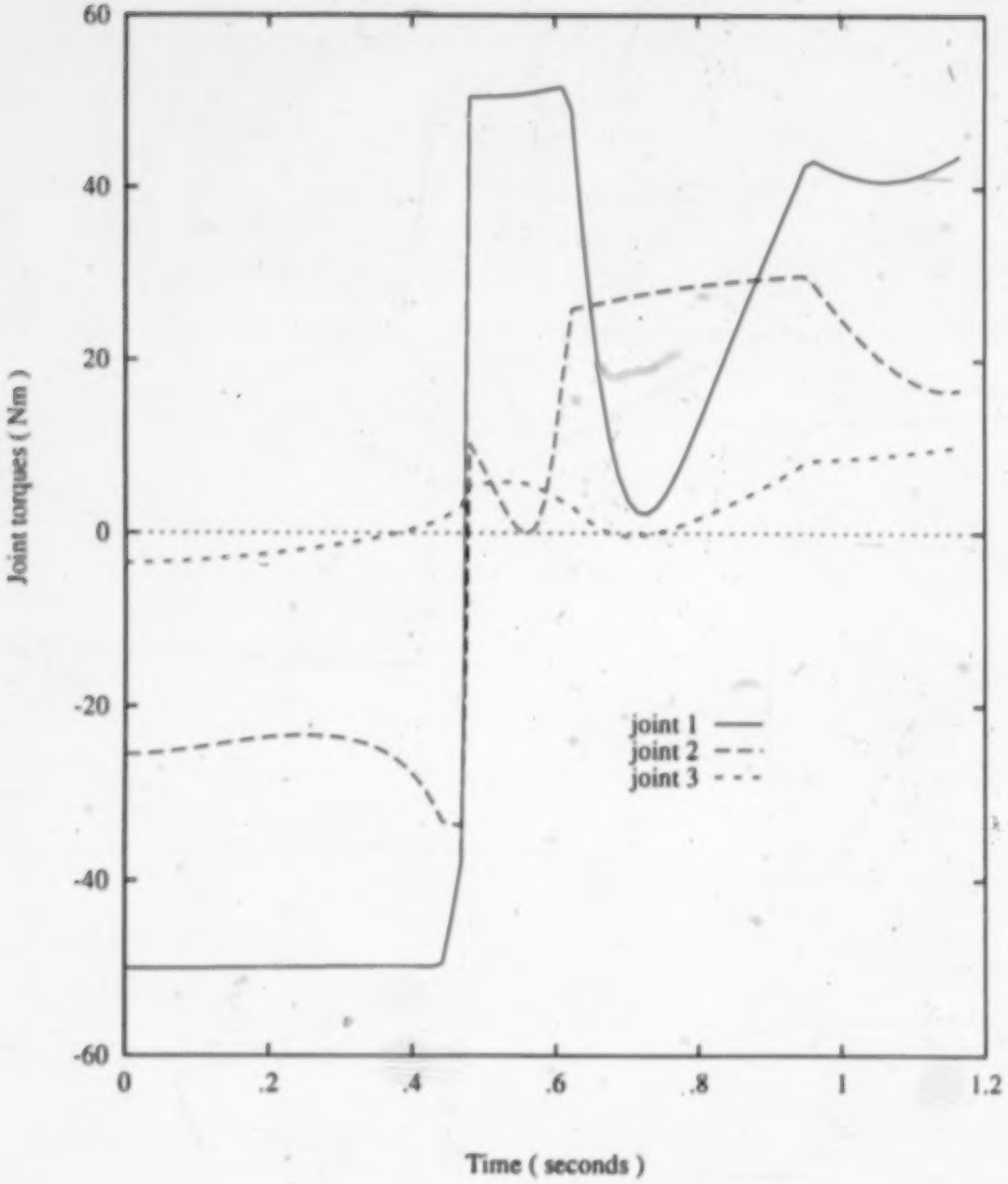
60
40
20
0
-20
-40
-60
0
.2
.4
.6
.8
1
1.2
Joint torques (Nm)
Time (seconds)
joint 1
joint 2
joint 3

Figure 5.6 (c)

Chapter 6

Computer Implementation

6.1 Introduction

The time optimal control and the obstacle avoidance algorithms discussed in chapters 4 and 5 are implemented on the three link manipulator model given in chapter three. The algorithms are programmed in C language and run on a SUN 3/160 workstation. The coding of these algorithms are not included in this thesis. However, it is available from in the Robotics Laboratory of the Electrical Engineering department. In this chapter, the structure of the programs and the subroutines which implement the minimum time and obstacle avoidance algorithms are described. This is essentially the computation of the required torque that drives the arm from a given initial position to the final position. In the last section of this chapter, the same three link planar redundant manipulator is simulated and the calculated torque is used to drive the simulated manipulator to verify the results obtained in chapters four and five.

6.2 Program Structure

The main program is built up by linking together various files. Each file is designated to handle one aspect of the manipulator algorithm. The kinematics, dynamics, matrices manipulation, integration methods, trajectory generation, optimal time algorithm, and the obstacle avoidance algorithm. The inputs outputs are all done in

different files. All the necessary informations for solving the differential equation are being read in from one input file and all the outputs are printed into files, both are done in the same directory. The foregoing discussion outlines the structure of the major subroutines used for the calculation of the optimal time torque with or without obstacle.

6.2.1 Time Optimal Algorithm

In the course of the formulation of the time optimal control problem, it was transformed to a second order differential acceleration equation $\ddot{x}=g(x,\dot{x})$ and deceleration equation $\ddot{x}=f(x,\dot{x})$. The solution to the time optimal problem is virtually solving these two initial boundary value problems. The algorithm and technique for solving a second order differential equation has been given in details in chapter four. The computer computation of the equations given earlier in chapter four is presented here.

The function of $g(x,\dot{x})$ and $f(x,\dot{x})$ are calculated by the subroutine called GF. The arguments are x, $\dot{x}$, $\ddot{x}$ and an input flag for the choice for the maximum acceleration or deceleration. The subroutine takes x and $\dot{x}$ as inputs and passes the results $\ddot{x}$ back to the calling function as an argument. Note that the joint angle, and velocities are not passed as arguments because they are defined as global variables to all the others subroutines in the same file. In addition, *GF* is eventually called by the differential equation solver subroutine which consider *GF* dependent on x and $\dot{x}$ only. From equations (4.9.2) and (4.9.3), $g(x,\dot{x})$ and $f(x,\dot{x})$ are calculated from $T_{i_}$, $T_{i_}$, C_{1i} and C_{2i}. The maximum and minimum allowable torques are calculated from the subroutine *Toqbd* which has the maximum and minimum torque vectors as argument. The torque boundary used in this implementation is taken as functions of the joint velocities only, it is calculated as

$$T_{i_} = \ddot{\theta} + T_i \tag{6.1.1}$$

$$T_{i_} = \dot{\theta} - T_i \tag{6.1.2}$$

The evaluation of the C_{1i} and C_{2i} is more complicated and is done in the subroutine $C12$. Only x and $\dot{x}$ are passed as arguments, the output vectors $C1$ and $C2$ are declared global. In $C12$, the previous joint angle vector *thest* which is global to all the files are first copied to the current joint angle vector *the* as the starting angles. The starting angle is crucial to the evaluation of the inverse kinematics. The joint angles and velocities are calculated in *ang_vel* which takes x, $\dot{x}$ as input arguments and the joint angle and velocity vectors as output arguments. Equations (4.11) and (4.6.3) with the pseudoinverse replacing the Jacobian inverse are evaluated in *ang_vel*. In *ang_vel*, subroutine *del_pth* is then called to find the change in the Cartesian position of the end effector at the displacement x. *del_pth* takes the previous x and finds the Cartesian position of the end effector at x by calling another subroutine *path*. *path* can generate straight line paths in different directions taking the displacement x as input and evaluate the Cartesian position from the initial Cartesian position and the path direction. The choice of the path direction is specified in the input file. The initial Cartesian and joint positions are global to all files. *del_pth* and *path* are written in a separate file for they define and calculate the trajectory position and its change.

With $d\mathbf{r}$ evaluated by *del_pth*, $d\theta$ in (4.11.2) can now be found. The joint angles corresponding to the displacement x are calculated by *Cart_jt* which evaluates the pseudoinverse of the Jacobian of equation (4.11.2) at the previous joint angles and multiply it with the output of *del_pth*. The joint angles at x are finally obtained by adding $d\theta$ to the old joint angles. The new joint angles are used for the calculation of the joint velocity. The unit vector v, tangential to the path is found by subroutine *Vect_rx*. Consider the end effector being perturbed by a small displacement along the

trajectory. The difference between the perturbed vector and the original position vector gives the tangential vector along the trajectory. *Vect_rx* calculates the tangential vector and normalizes it to a unit vector. Now, the calculation of the joint velocity vector can be done easily as in (4.6.3).

The computation of C_1 requires the evaluation of the inertia matrix and the Jacobian at the joint angles found by *ang_vel*. The inertia matrix and the Jacobian are both given in chapter three. Using the previously computed **r** in equation (4.8.2), it is then straight forward to obtain C_1.

The computation of C_2, however is much more complicated than that of c_1 because it involves taking the derivative of the Jacobian. Similar to the Jacobian, its time derivative is first calculated analytically and then implemented on the program. By chain rule, the analytical expression for the derivative of the Jacobian can be written as

$$\frac{d}{dt} J(\theta) = \frac{dJ(\underline{\theta})}{d\underline{\theta}} \frac{d\theta}{dt} \tag{6.2.1}$$

$$= \frac{\partial J(\theta)}{\partial \theta_1}\dot{\theta}_1 + \frac{\partial J(\theta)}{\partial \theta_2}\dot{\theta}_2 + \frac{\partial J(\theta)}{\partial \theta_3}\dot{\theta}_3 \tag{6.2.2}$$

where each term in (4.13.4) represents a matrix. The term $\dot{J}(\theta)$ can be expressed using the same notations for the sine and cosine terms given in chapter 3 as follow

$$\begin{bmatrix} -l_1 c_1\dot{\theta}_1 - l_2 c_{12}(\dot{\theta}_1+\dot{\theta}_2) - l_3 c_{123}(\dot{\theta}_1+\dot{\theta}_2+\dot{\theta}_3) & -l_2 c_{12}(\dot{\theta}_1+\dot{\theta}_2) - l_3 c_{123}(\dot{\theta}_1+\dot{\theta}_2+\dot{\theta}_3) & -l_3 c_{123}(\dot{\theta}_1+\dot{\theta}_2+\dot{\theta}_3) \\ -l_1 s_1\dot{\theta}_1 - l_2 s_{12}(\dot{\theta}_1+\dot{\theta}_2) - l_3 s_{123}(\dot{\theta}_1+\dot{\theta}_2+\dot{\theta}_3) & -l_2 s_{12}(\dot{\theta}_1+\dot{\theta}_2) - l_3 s_{123}(\dot{\theta}_1+\dot{\theta}_2+\dot{\theta}_3) & -l_3 s_{123}(\dot{\theta}_1+\dot{\theta}_2+\dot{\theta}_3) \end{bmatrix} \tag{6.3}$$

Having both C_1 and C_2 computed, all the terms in equation (4.9.1) and (4.9.2) are known. According to the input flag, the maximum acceleration and the maximum deceleration can be chosen for forward and backward integration. Finally, the maximum of f_i and the minimum of g_i are found by subroutines *maximum* and *minimum* respectively. The the maximum of f_i or the minimum of g_i is passed to the calling function as $\ddot{x}$. This completes the description of the subroutine *GF*.

6.2.2 Obstacle Avoidance Algorithm

The computer implementation of the obstacle avoidance algorithm follows closely the derivation given in chapter five. Equations (5.4) to (5.9) are used to find derivative of the Jacobian. Subroutines *inv_fcn* is written to find the inverse function solution for the inverse kinematics, it takes the the time, joint angles, the displacement along the trajectory x and *del_pth* as inputs and replaces the old joint angles by the new one. The required constants are read from the input file. Subroutine *mk_Pf* is also written to find the Jacobian given in equation (5.6) and is being called in *inv_fcn* to compose the new Jacobian numerically. Subroutine *obs_ang_vel* is also written to replace *ang_vel* to include the obstacle avoidance algorithm for finding the joint angles and velocities upon calling *mk_Pf* and *inv_fcn*. Subroutine *C*12 is also modified as *obs_c*12 to find the corresponding *C*1 and *C*2 terms. The major modification is given in equation (5.10).

6.3 Simulation and Results

The computer simulation of the model of the three link redundant manipulator given in chapter three is discussed in detail here. The kinematics and dynamics equations derived earlier are implemented to verify the calculated torques in the previous chapters. The computer simulated redundant manipulator is basically a program which

takes a set of joint torques as inputs and generates a set of joint angles as outputs. The redundant manipulator is also animated by another graphic program that takes the joint angles as inputs and plots the corresponding postures on the screen at constant time intervals.

To simulate the motion of the redundant manipulator, the dynamics of the manipulator must be used. The same assumptions and dynamics as in chapter three are used here for computer implementation of the manipulator. Having all the relevant terms in the dynamic equation (1.5) derived in closed form, computer simulation of the dynamics by iterations is not taken into consideration. The inverse dynamics is solved instead to simulated the movement of the manipulator. That is, the joint accelerations are expressed in terms of the joint torques as

$$\ddot{\theta} = M^{-1}(\theta)(\tau - V(\theta,\dot{\theta})) \tag{6.4}$$

where the gravitation term and the friction term are dropped because of the assumption stated earlier. Numerical evaluation of equation (6.1) is straight forward with M and V fully defined. The inversion of the mass inertia matrix does not require special attention in general because of its positive definite property. The numerical inversion of a matrix is done by the well known lower triangle upper triangle (LU) decomposition and back substitution. Although the Singular Value Decomposition method mentioned in chapter two can also be used to find the normal inverse, it is used only to find the pseudoinverse of the kinematics in the formulation of the time optimal and obstacle problem. The joint accelerations can be found according to equation (6.4) with one matrix inversion, multiplication and subtraction.

The difficulties of this computer simulation process lies in the evaluation of the joint angles. The inverse dynamics equation has to be integrated forward in time with

given initial joint positions and velocities. Equation (6.4) can either be treated as a system of second order ordinary differential equation or it can be reduced to a first order system of differential equation by the state space approach. It is obviously easier to handle a first order system than a second order one. Denote $\mathbf{y}_1$ and $\mathbf{y}_2$ as θ and $\dot{\theta}$ respectively, the above equation can be rewritten as a system of first order differential equations

$$\begin{bmatrix} \dot{\mathbf{y}}_1 \\ \dot{\mathbf{y}}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{y}_2 \\ M^{-1}(\theta)(\tau - V(\theta,\dot{\theta})) \end{bmatrix} \tag{6.5}$$

There are actually six first order differential equations in (6.2) for each $\mathbf{y}$ contains three equations. The six equations are non-linear and strongly coupled with each other. The joint angles can be solved theoretically by any initial value differential equation solver, however, the integration technique mentioned earlier is used.

The motion produced by the joint torques given in Figure 4.6 (c), Figure 4.9 (c) and Fig.5.6 (c) are used as input torques to the simulation program and the resulting motions are shown in Figure 6.1, Figure 6.2 and Figure 6.3 respectively. Comparing Figure 6.1, Figure 6.2, and Figure 6.3 with Figure 4.4, Figure 4.7 and Figure 5.4 respectively, hardly any difference can be noted, however looking at the numerical values, there is a slight error in the simulation result.

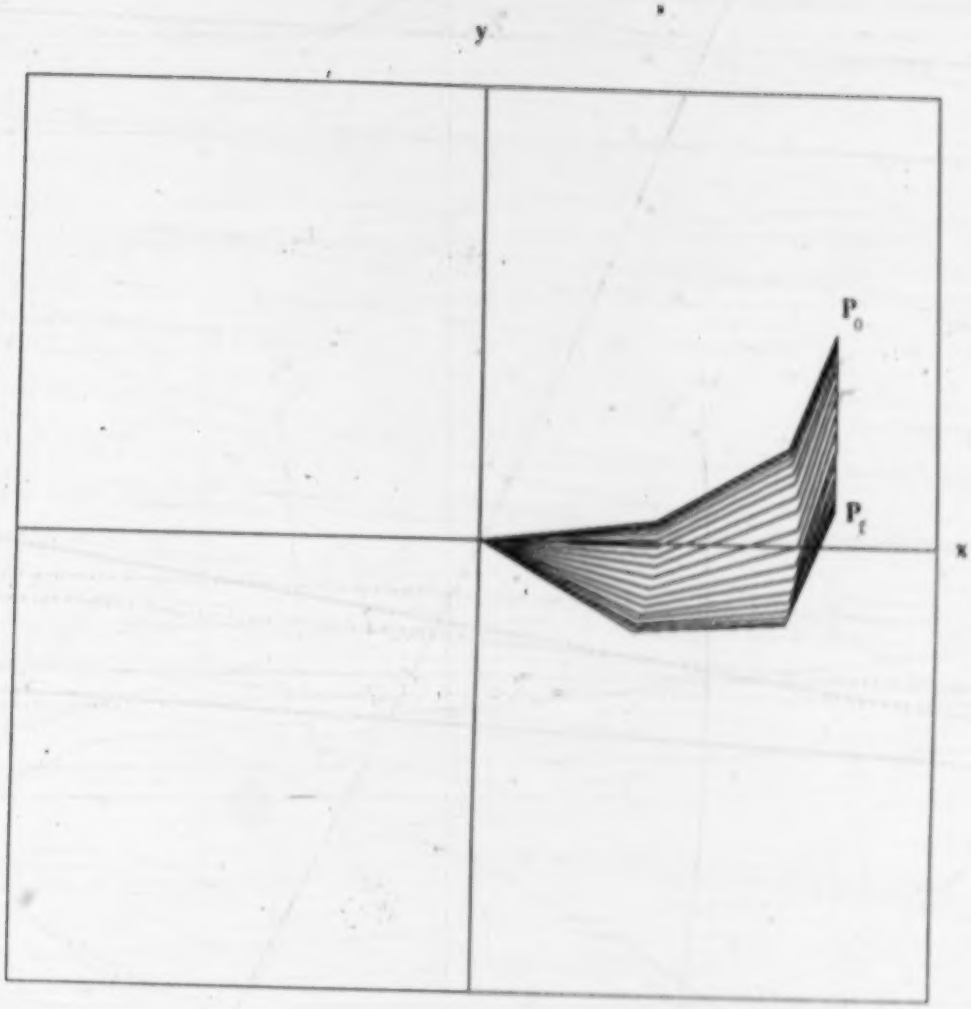

Figure 6.1 Simulation Result of Figure 4.4

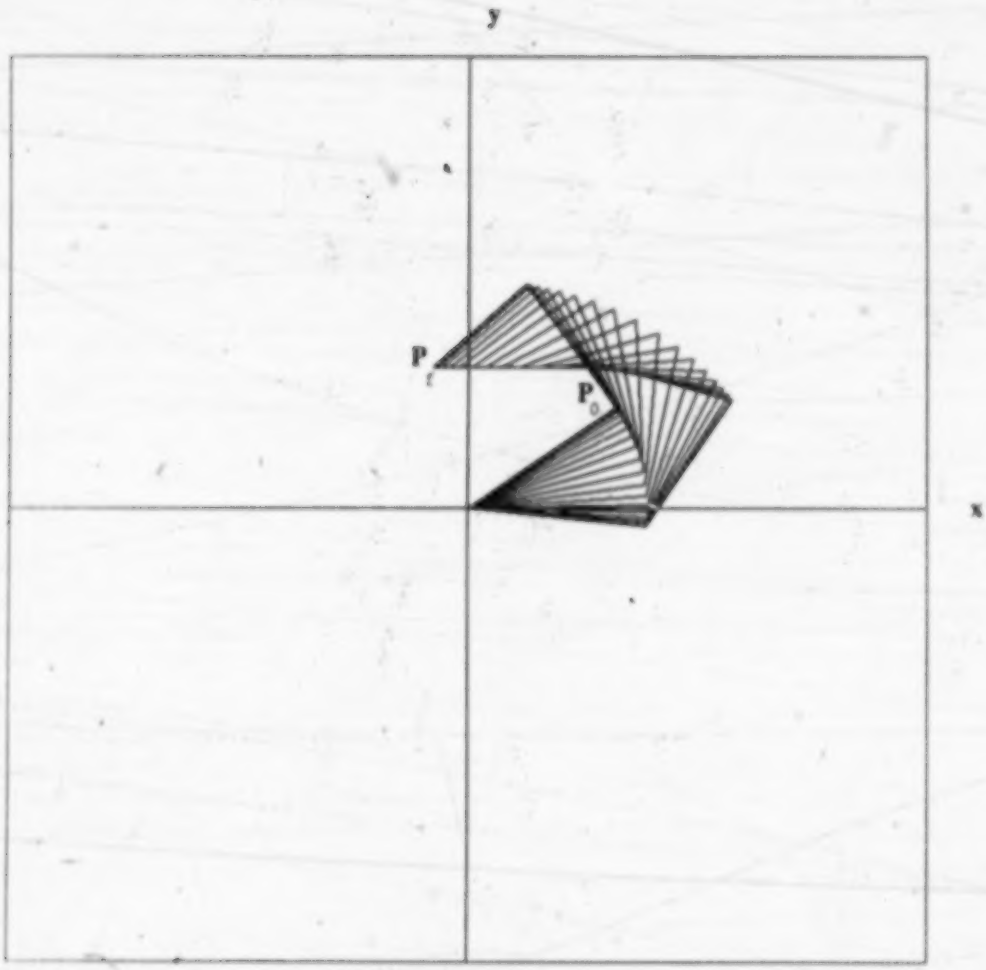

Figure 6.2 Simulation Result of Figure 4.7

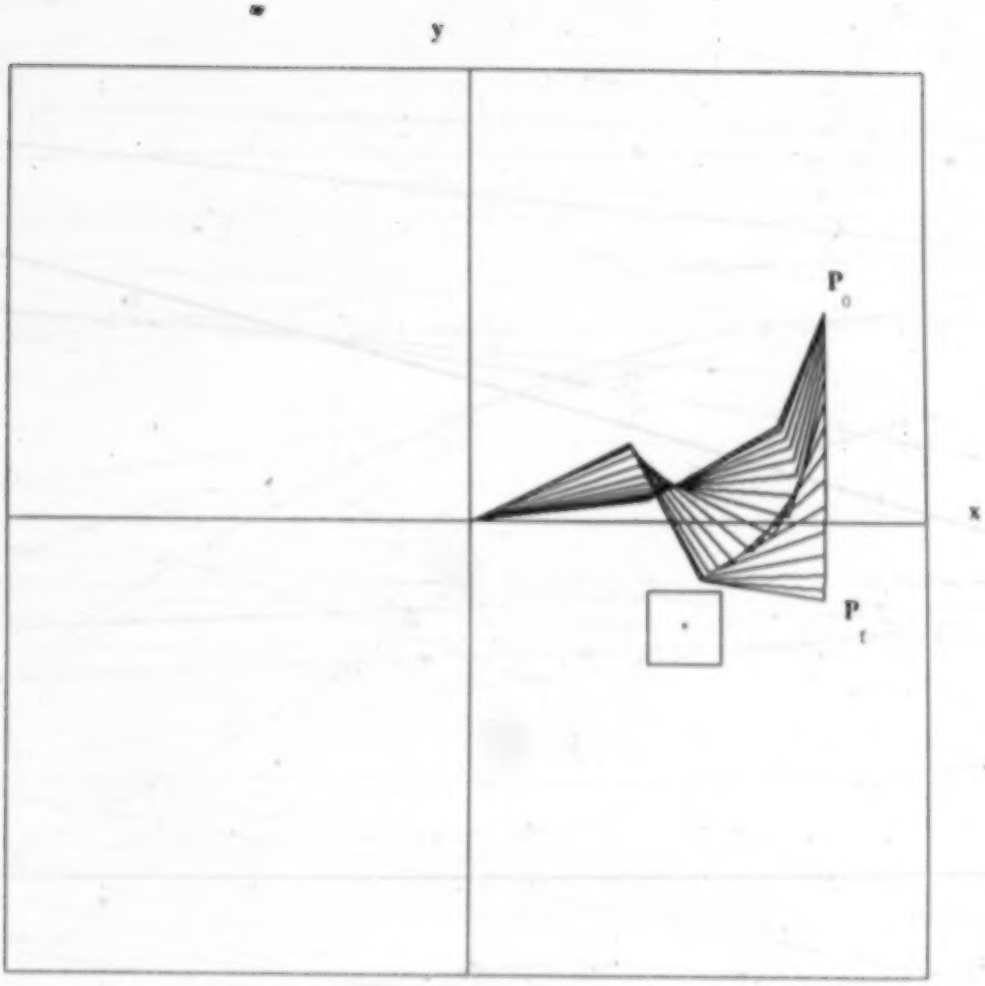

Figure 6.3 Simulation Result of Figure 5.4

Chapter 7

Conclusion

A time optimal algorithm for a redundant manipulator has been presented and implemented on a three link planar manipulator. This algorithm is an extension of the non-redundant minimum time algorithm [Bobrow 83] to the redundant case. This algorithm uses the displacement along the trajectory in the Cartesian plane as the control variable and obtain the minimum time of a specified trajectory. The pseudoinverse is used to find a unique inverse kinematics solution out of all the possibilities that satisfy the end effector requirement. The inverse kinematics solution is obtained by evaluating the increments of all the joint angles through the pseudoinverse and adding them to the previous joint angles. Given the initial joint angles, the minimum time for the specified trajectory is found. Examples presented in this thesis required single switching only. It has been very difficult to find a multiple switching case to illustrate the multiple switching algorithm, thus multiple switching examples are not presented.

Time optimal solutions for specified path or obstacle avoidance has been developed as seen from the literature for the non-redundant case. However, a minimum time obstacle avoidance algorithm for a specified path has been given here with the use of a redundant manipulator. This obstacle avoidance algorithm is developed in the light of the inverse function approach [Wampler 87]. Instead of the conventional null space approach for obstacle avoidance [Nakamura et el. 81], an additional equation is added to the underdetermined kinematics equations to fully define the system.

The function used is the sum of the distance to a constant power between the center of the obstacle and each individual joint. Obstacle avoidance is achieved by setting the gradient of this function by choosing an appropriate constant. This algorithm is then incorporated to the time optimal algorithm to get the minimum time obstacle avoidance trajectory.

The presented algorithms are not limited to the planar three link redundant manipulator, it can be extended to a three dimensional seven or eight degrees of freedom redundant manipulator. The obstacle avoidance algorithm can also be extended to include more than one obstacle for manipulators with more than one degree of redundancy.

BIBLIOGRAPHY

1. Albert, A., *Regression and the Moore-Penrose Pseudoinverse*, Academic Press, New York, 1972.

2. Birkhoff, G. and Rota, G., *Ordinary Differential Equations, Blaisdell, New York, 1969.*

3. Bobrow, J.E., Dubowsky, S., and Gibson, J.S., *"On the Optimal Control of Robotic Manipulators with Actuator Constraints,"* Proc. American Control Conference, pp. 782-787, San Francisco, California, June 22-24, 1983.

4. Boullion, T.L. and Odell P.L., *Generalized Inverse Matrices*, Wiley-Interscience, New York, 1971.

5. Bulirsch, R. and Stoer, J., *"Asymptotic Upper and Lower Bounds for Results of Extrapolation Methods,"* Numerische Mathematik , vol. 8, pp. 93-104, 1966.

6. Dubey, R.V., Euler, S.M., Babcock, S.M., and Glassell, R.L., *"Real-time Implementation of a Kinematic Optimization Scheme for Seven-Degree-of-Freedom Redundant Robots with Spherical Wrists,"* Proc. IEEE Int. Conference on Robotics and Automation, pp. 1376- 1378, Philadelphia, April 1988.

7. Fatunla, S.O., *Numerical Methods for Initial Value Problems in Ordinary Differential Equations*, Academic Press, Inc., San Diego, 1988.

8. Featherstone, R., *"Position and Velocity Transformations between Robot End-Effector Coordinates and Joint Angles,"* Int'l Journal of Robotics Research, vol. 2, no. 2, pp. 35-45, 1983.

9. Gear, C.W., *Numerical Initial Value Problems in Ordinary Differential Equations*, Prentice Hall, New Jersey, 1971.

10. Geering, H., Guzzella, L., Hepner, S.R., and Onder, C.H., *"Time-Optimal Motions in Assembly Tasks,"* IEEE Transactions on Automatic Control, vol. AC-31, no. 6, pp. 512-518, June 1986.

11. Hollerbach, J. and Suh, K.C., *"Redundancy Resolution of Manipulators through Torque Optimization,"* Proc. IEEE Int. Conference on Robotics and Automation, pp. 1016- 1021, St. Louis , March 25-28, 1985.

12. Hollerbach, J., *"Optimum Kinematic Design for a Seven Degree of Freedom Manipulator,"* 2nd Int'l Symp. on Robotics Research, Kyoto, Japan, Aug. 20-23, 1984.

13. Hollerbach, J., Baillieul, J., and Brockett, R., "*Programming and Control of Kinematically Redundant Manipulators,*" Proceedings of 23rd Conference on Decision and Control, Las Vagas, NV, Dec 1984a.

14. Khatib, O. and Le Maitre, J.F., "*Dynamic control of manipulators operating in a complex environment,*" Proc. 3rd Int. CISM-IFToMM Symp., pp. 267-282, Udine, Italy, 1978.

15. Kim, B.K. and Shin, K.G., "*Minimum-Time Path Planning for Robot Arms and Their Dynamics,*" IEEE Trans. Syst., Man, Cybern,, vol. SMC-15, no. 6, pp. 213-223, Mar./Apr. 1985.

16. Klein, C.A. and Huang, C.H., "*Review of Pseudoinverse Control for Use with Kinematically Redundant Manipulators,*" IEEE Trans. Syst., Man, Cybern,, vol. SMC-13, no. 3, pp. 245-250, Mar./Apr. 1983.

17. Lambert, J.D., *Computational Methods in ODE's*, John Wiley, New York, 1973.

18. Nakamura, Y., Hanafusa, H., and Yoshikawa, T., "*Analysis and Control of Articulated Robot Arms With Redundancy,*" Prep. 8th IFAC World Congress, XIV, pp. 78-83, 1981.

19. Nakamura, Y., Hanafusa, H., and Yoshikawa, T., "*Task- Priority Based Redundancy Control of Robot Manipulators,*" The Int. Journal of Robotics Research, vol. 6, no. 2, M.I.T, Summer 1987.

20. Paul, R. and Stevenson, C.N., "*Kinematics of Robot Wrists,*" Int'l Journal of Robotics Research, vol. 2, no. 1, pp. 31-38, 1983.

21. Shiller, Z., Norris, M.A., and Dubowsky, S., "*Time Optimal Path Planning for Robotic Manipulators with Obstacle Avoidance: A CAD Approach,*" Proc. IEEE Int. Conference on Robotics and Automation, San Francisco, CA, March 1986.

22. Shiller, Z. and Lu, H.H., "*Robust Computation of Path Constrained Time Optimal Motions,*" Proc. IEEE Int. Conference on Robotics and Automation, pp. 144-149, 1990.

23. Shiller, Z., Bobrow, J.E., Dubowsky, S., and Gibson, J.S., "*Time Optimal Control of Robotic Manipulators,*" Proc. IEEE Int. Conference on Robotics and Automation, vol. 4, no. 3, pp. 3-17, 1985.

24. Shiller, Z. and Dubowsky, S., "*Time Optimal Paths and Acceleration Lines of Robotic Manipulators,*" Proc. of IEEE Transactions on Automatic Control, vol. AC-30, no. 6, pp. 531-541, June 1985.

25. Shiller, Z. and Dubowsky, S., "*On the Optimal Trajectories for Robotic Manipulator with Actuators and End Effector Constraints,*" Proc. IEEE Int. Conference on Robotics and Automation, St. Louis, March, 1985.

26. Shin, K.G. and McKay, N.D., "*Minimum-Time Control of Robotic Manippulators with Geometric Path Constraints*," IEEE Transactions on Automatic Control, vol. AC-30, no. 6, pp. 531-541, June 1985.

27. Takase, K., Inoue, H., and Sato, K., "*The Design of an Articulated Manipulator with Torque Control Ability*," Proc. 4th Int'l. Symp. Industrial Robots , pp. 261-270, Nov 19-24, 1974.

28. Trevelyan, J.P., Kovesi, P.D., and Ong, M.C.H., "*Motion Control for a Sheep Shearing Robot*," Robotic Research : The First Int'l Symposium, pp. 175-190, M.I.T. Press, Cambridge, Mass, 1984.

29. Wampler, C.W., "*The Inverse Function Approach to Kinematic Control of Redundant Manipulator*," Proc. IEEE Int. Conference on Robotics and Automation, pp. 1364- 1369, Philadelphia, April 1988.